Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN LICENSE.PDF) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document contains confidential information that is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, Two-Second Advantage, TIB, TIBCO Adapter, Predictive Business, Information Bus, TIBCO BusinessConnect, TIBCO ActiveMatrix BusinessWorks, TIBCO Enterprise Message Service are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Enterprise Java Beans (EJB), Java Platform Enterprise Edition (Java EE), Java 2 Platform Enterprise Edition (J2EE), and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle Corporation in the U.S. and other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

THIS SOFTWARE MAY BE AVAILABLE ON MULTIPLE OPERATING SYSTEMS. HOWEVER, NOT ALL OPERATING SYSTEM PLATFORMS FOR A SPECIFIC SOFTWARE VERSION ARE RELEASED AT THE SAME TIME. SEE THE README.TXT FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This Product is covered by U.S. Patent No. 7,472,101.

Copyright © 1999-2014 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information
Contents

Tables ...................................................................................................................... vii

Preface ..................................................................................................................... ix
Changes from the Previous Release of This Guide .................................................... x
Related Documentation .............................................................................................. xi
TIBCO MDM Documentation ...................................................................................... xi
Other TIBCO Product Documentation ...................................................................... xi
Typographical Conventions ...................................................................................... xiii
Connecting with TIBCO Resources ........................................................................... xiv
How to Join TIBCOmunity ........................................................................................ xiv
How to Access TIBCO Documentation ..................................................................... xiv
How to Contact TIBCO Support ................................................................................ xiv

Chapter 1 Overview ................................................................................................. 1
Introduction ............................................................................................................... 2
Working of Web Services in TIBCO MDM ................................................................. 2
Web Service Message Structure ............................................................................... 5
Generic Schema Based Message Structure .............................................................. 5
Strict Schema Based Message Structure ..................................................................... 8
Samples ....................................................................................................................... 9
Accessing Web Services ............................................................................................. 10
Validation of XML ...................................................................................................... 12
Validation of XML Instances ................................................................................... 12
Including Namespace in Response ............................................................................. 16
Design Principles ....................................................................................................... 17
Authentication ........................................................................................................... 18
Administrator Web Services ...................................................................................... 20
Data Web Services .................................................................................................... 22
DataLoad Web Services ............................................................................................. 23
Content Web Services ............................................................................................... 23
Event Management Web Services .............................................................................. 25
Faults .......................................................................................................................... 26
Administrator Web Services ...................................................................................... 26
Data Web Services .................................................................................................... 28
DataLoad Web Services ............................................................................................. 29
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Web Services</td>
<td>31</td>
</tr>
<tr>
<td>Event Management Web Services</td>
<td>32</td>
</tr>
<tr>
<td>Password Encryption and Decryption</td>
<td>33</td>
</tr>
<tr>
<td>Session Handling</td>
<td>35</td>
</tr>
<tr>
<td>Web Services Available in TIBCO MDM</td>
<td>39</td>
</tr>
<tr>
<td>Web Services Accepting Multiple Entities</td>
<td>44</td>
</tr>
<tr>
<td><strong>Chapter 2  Record Services</strong></td>
<td>47</td>
</tr>
<tr>
<td>CRUD Services for Record Services</td>
<td>48</td>
</tr>
<tr>
<td>Repositories with FED attributes</td>
<td>49</td>
</tr>
<tr>
<td>Generating WSDL</td>
<td>49</td>
</tr>
<tr>
<td>WSDL and XML Schema Generation</td>
<td>54</td>
</tr>
<tr>
<td>CRUD Operations</td>
<td>56</td>
</tr>
<tr>
<td>Record Query Service</td>
<td>61</td>
</tr>
<tr>
<td>Record Query Metadata Service</td>
<td>84</td>
</tr>
<tr>
<td>Record Query ValidValues (Init) Service</td>
<td>88</td>
</tr>
<tr>
<td>Record Validate Service</td>
<td>94</td>
</tr>
<tr>
<td>Record Add or Modify Services</td>
<td>96</td>
</tr>
<tr>
<td>Record Delete Service</td>
<td>105</td>
</tr>
<tr>
<td>Initiate Workflow Service</td>
<td>109</td>
</tr>
<tr>
<td>Transaction Atomicity for Record Web Services</td>
<td>112</td>
</tr>
<tr>
<td>Scenario 1: Process Multiple CUD Commands</td>
<td>114</td>
</tr>
<tr>
<td>Scenario 2: Large Number of Modified Records</td>
<td>115</td>
</tr>
<tr>
<td>Scenario 3: Enable Global Distributed Locking</td>
<td>115</td>
</tr>
<tr>
<td><strong>Chapter 3  Work Item Services</strong></td>
<td>117</td>
</tr>
<tr>
<td>Work Item Query Parameters Service</td>
<td>118</td>
</tr>
<tr>
<td>Work Item Query Service</td>
<td>120</td>
</tr>
<tr>
<td>Work Item Modify Close Service</td>
<td>126</td>
</tr>
<tr>
<td>Work Item Lock Service</td>
<td>130</td>
</tr>
<tr>
<td>Work Item UnLock Service</td>
<td>132</td>
</tr>
<tr>
<td>Get Work Item Service</td>
<td>134</td>
</tr>
<tr>
<td>Work Item Reassign Service</td>
<td>140</td>
</tr>
<tr>
<td>Get Custom Work Item Summary Service</td>
<td>142</td>
</tr>
<tr>
<td>Get Predefined Work Item Summary Service</td>
<td>148</td>
</tr>
<tr>
<td><strong>Chapter 4  Administration Services</strong></td>
<td>151</td>
</tr>
<tr>
<td>Login Service</td>
<td>152</td>
</tr>
</tbody>
</table>
## Contents

- **Logout Service** .......................................................... 153
- **Get Entitlement Service** ................................................. 154
- **Create Enterprise Service** ............................................. 157
- **Get Enterprise List Service** .......................................... 159
- **Create User Service** ................................................... 161
- **Modify User Service** .................................................. 166
- **Change Password Service** ........................................... 172
- **Delete User Service** .................................................. 174
- **Get User List Service** ................................................ 176
- **Get Role List Service** ................................................ 178

**Chapter 5 Data Services** .................................................. 181

- **Text Search Service** .................................................. 182
  - Enabling Text Search ................................................. 183
  - Types of Text Search Services ................................. 184
- **Get DataSource List Service** ........................................ 191
- **Get Repository List Service** ....................................... 193
- **Get Related Repository List Service** .......................... 195
- **Get Repository with Dependant Artifacts Service** ........ 197
- **Initiate Synchronization Service** .............................. 199
- **Upload Metadata Service** .......................................... 201
- **Extract Data Service** ................................................ 203
  - Quick Export Service Using Data Extractor ............... 205
- **GetClassificationSchemeList** ................................... 210
- **GetClassificationSchemeAttributeList** .................... 213
- **GetClassificationCodeHierarchy** ............................... 216

**Chapter 6 DataLoad Services** ......................................... 225

- **Get DataSource Info Service** .................................... 226
- **Upload DataSource Service** ....................................... 228
- **Import Records Service** .......................................... 230
- **Load Import Records Service** ................................... 232

**Chapter 7 Content Services** .......................................... 235

- **Get Content Service** ............................................... 236
Chapter 8  Event Management Services ................................. 239
Get Event Info Service .......................................................... 240
Get Event Details Service ...................................................... 242
Get Import Event Status Service ............................................. 244

Index .................................................................................. 247
Tables

Table 1  General Typographical Conventions ................................................................. xiii
Table 2  Elements in a Web Service Request or Response .................................................. 5
Table 3  Endpoint Dispatching .......................................................................................... 10
Table 4  SOAP Action Dispatching .................................................................................... 11
Table 5  Attributes of Payload.xsd ................................................................................... 14
Table 6  Authentication of Administrator Web Services ...................................................... 20
Table 7  Authentication of Data Web Services .................................................................... 22
Table 8  Authentication of DataLoad Web Services .............................................................. 23
Table 9  Authentication of Content Web Services ................................................................. 24
Table 10 Authentication of Event Management Web Services ............................................. 25
Table 11 Faults of Administrator Web Services ................................................................. 26
Table 12 Faults of Data Web Services ............................................................................... 28
Table 13 Faults of DataLoad Web Services ........................................................................ 29
Table 14 Faults of Content Web Services .......................................................................... 31
Table 15 Faults of Event Management Web Services ......................................................... 32
Table 16 UI Browser Web Client ....................................................................................... 35
Table 17 Web Services Available in TIBCO MDM ............................................................... 39
Table 18 Web Services Accepting Multiple Entities ............................................................ 44
Table 19 XSD Types .......................................................................................................... 55
Table 20 Record Query - External Keys ............................................................................ 63
Table 21 Record Query - Supported Operators ................................................................... 69
Table 22 Record Query - Deprecated and Equivalent Operators ....................................... 70
Table 23 Record Query - Keys ........................................................................................... 72
Table 24 Relationship Type for Perspective and Description ............................................... 81
Table 25 Record Query MetaData - External Keys ............................................................... 84
Preface

The TIBCO® MDM is a tool to manage master data of your organization by providing a framework for governance, rules, and processes.

This ensures accuracy and efficiency both inside the enterprise as well as throughout the value chain so that multiple processes are optimally coordinated. TIBCO MDM delivers a multi domain horizontal platform to manage all types of information including products, customers, vendors, reference data, trading partners, and so on.

Topics

- Changes from the Previous Release of This Guide, page x
- Related Documentation, page xi
- Typographical Conventions, page xiii
- Connecting with TIBCO Resources, page xiv
This section itemizes the major changes to web services from the previous release:

**Updated Web Services**

The following web services are enhanced in this release:

Data Services

Fuzzy search against a child repository with `join=false` and related index entity defined on the child repository

- Text Search Service, page 182

Work Item Web Services

The `ReasonCode` tag is deprecated from the Work Item Modify Close web service.

- Work Item Modify Close Service, page 126
Related Documentation

This section lists documentation resources you may find useful.

TIBCO MDM Documentation

The following documents form the TIBCO MDM documentation set:

- *TIBCO MDM Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.
- *TIBCO MDM Installation and Configuration Guide*: Read this manual for instructions on site preparation, installation, and configuration.
- *TIBCO MDM User’s Guide*: This manual explains features and benefits of TIBCO MDM from the business user’s viewpoint. It describes the features and functionality as well as all the screens.
- *TIBCO MDM System Administration*: This manual explains features relevant to the system administrator.
- *TIBCO MDM Customization*: Read this manual to how the application can be customized to your enterprise needs.
- *TIBCO MDM Workflow Reference*: This manual is a reference for automation of business processes.
- *TIBCO MDM Web Services Guide*: This manual is a reference for using web services in TIBCO MDM.
- *JAVA API Reference*: This Help includes a list of workflows that are used in TIBCO MDM.
- *TIBCO MDM Best Practices Guide*: This manual provides the best practices based on contributions from the users of TIBCO MDM, who develop the software and implement it in a variety of TIBCO MDM projects.
- *TIBCO MDM Performance Tuning Guide*: This manual provides the performance tuning methodologies to tune your system and to achieve the optimal system performance on all of the layers of TIBCO MDM.

Other TIBCO Product Documentation

You may find it useful to read the documentation for the following TIBCO products:
- **TIBCO MDM Studio Installation Guide**: Read this manual for instructions on installation of TIBCO MDM Studio.

- **TIBCO MDM Studio Process Designer User’s Guide**: This guide is a reference for designing workflows using the TIBCO MDM Process Designer graphical user interface.

- **TIBCO MDM Studio Process Designer Tutorial**: This guide is a tutorial for designing workflows using the TIBCO MDM Process Designer graphical user interface.

- **TIBCO MDM Studio Repository Designer User’s Guide**: This guide is a reference for designing repositories using the TIBCO MDM Repository Designer graphical user interface.

- **TIBCO MDM Studio Repository Designer Tutorial**: This guide is a tutorial for designing repositories using the TIBCO MDM Repository Designer graphical user interface.

- **TIBCO MDM Studio Rulebase Designer User’s Guide**: This guide is a reference for designing rulebases using the TIBCO MDM Rulebase Designer graphical user interface.

- **TIBCO MDM Studio Rulebase Designer Tutorial**: This guide is a tutorial for designing rulebases using the TIBCO MDM Rulebase Designer graphical user interface.

- **TIBCO Enterprise Message Service™ software**: This software allows the application to send and receive messages using the Java Message Service (JMS) protocol. It also integrates with TIBCO Rendezvous and TIBCO SmartSockets™ messaging products.

- **TIBCO BusinessWorks software™**: This is a scalable, extensible and easy to use integration platform that allows you to develop and test integration projects. It includes a graphical user interface (GUI) for defining business processes and an engine that executes the process.

- **TIBCO BusinessConnect™ software**: This software allows your company to send and receive XML or non-XML business documents over the Internet. Based on a mutually agreed process flow and common document format, you and your trading partners can conduct secure and verifiable business transactions online.
### Typographical Conventions

The following typographical conventions are used in this manual.

**Table 1  General Typographical Conventions**

<table>
<thead>
<tr>
<th>Convention</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>code font</td>
<td>Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example: Use MyCommand to start the foo process.</td>
</tr>
<tr>
<td>bold code font</td>
<td>Bold code font is used in the following ways:</td>
</tr>
<tr>
<td></td>
<td>• In procedures, to indicate what a user types. For example: Type admin.</td>
</tr>
<tr>
<td></td>
<td>• In large code samples, to indicate the parts of the sample that are of particular interest.</td>
</tr>
<tr>
<td></td>
<td>• In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, MyCommand is enabled: MyCommand [enable</td>
</tr>
<tr>
<td>italic font</td>
<td>Italic font is used in the following ways:</td>
</tr>
<tr>
<td></td>
<td>• To indicate a document title. For example: See TIBCO BusinessWorks Concepts.</td>
</tr>
<tr>
<td></td>
<td>• To introduce new terms For example: A portal page may contain several portlets. Portlets are mini-applications that run in a portal.</td>
</tr>
<tr>
<td></td>
<td>• To indicate a variable in a command or code syntax that you must replace. For example: MyCommand pathname</td>
</tr>
<tr>
<td>Key combinations</td>
<td>Key name separated by a plus sign indicate keys pressed simultaneously. For example: Ctrl+C.</td>
</tr>
<tr>
<td></td>
<td>Key names separated by a comma and space indicate keys pressed one after the other. For example: Esc, Ctrl+Q.</td>
</tr>
<tr>
<td>Note icon</td>
<td>The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.</td>
</tr>
<tr>
<td>Tip icon</td>
<td>The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.</td>
</tr>
<tr>
<td>Warning icon</td>
<td>The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.</td>
</tr>
</tbody>
</table>
Connecting with TIBCO Resources

How to Join TIBCOmmunity

TIBCOmmunity is an online destination for TIBCO customers, partners, and resident experts. It is a place to share and access the collective experience of the TIBCO community. TIBCOmmunity offers forums, blogs, and access to a variety of resources. To register, go to http://www.tibcommunity.com.

How to Access TIBCO Documentation

You can access TIBCO documentation here:

https://docs.tibco.com

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, contact TIBCO Support as follows:

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:
  http://www.tibco.com/services/support
- If you already have a valid maintenance or support contract, visit this site:
  https://support.tibco.com

Entry to this site requires a user name and password. If you do not have a user name, you can request one.
Chapter 1  Overview

This chapter introduces you to the web services in TIBCO MDM.

Topics

- Introduction, page 2
- Web Service Message Structure, page 5
- Accessing Web Services, page 10
- Validation of XML, page 12
- Including Namespace in Response, page 16
- Design Principles, page 17
- Authentication, page 18
- Faults, page 26
- Password Encryption and Decryption, page 33
- Session Handling, page 35
- Web Services Available in TIBCO MDM, page 39
- Web Services Accepting Multiple Entities, page 44
Introduction

Web services are self-contained modular applications that can be published, located, and invoked across the Web. Web services perform functions, which can be anything from simple requests to complicated business processes. Once a web service is deployed, other applications (and also other web services) can discover and invoke the deployed service.

Working of Web Services in TIBCO MDM

All web services in TIBCO MDM work on the request-response paradigm. Two parties can conduct electronic business by exchanging business documents. The sending party (Sender) packages one or more documents into a request XML message which is then sent to the receiving party (Receiver). The Receiver processes the message contents, generates a response XML, which is returned to the Sender that originated the request. The Sender is informed of the status (successful or otherwise) of the request message delivery.

Examples of the Sender’s documents could be purchase order requests, manufacturing information, and patient health care information. Examples of the Receiver’s responses include order confirmations, change control information, and contractual acknowledgements.

Execution Mode

A web service in TIBCO MDM can be executed in the Synchronous or Asynchronous mode. The web services that support these execution modes are Record Add, Record Modify, Delete Record, Initiate Workflow, and Close Work Item.

Synchronous Mode (SYNCHR)

With synchronous services, the client invokes a service and then waits for a response to the request.

In the Synchronous mode, when a record is saved for an action, a workflow start event is generated and the workflow event is sent for processing without sending the workflow event into JMS. The response waits for the workflow execution to end. Workflow execution may end when workflow completes or suspends waiting for a user input. Only after the workflow execution is completed, the following is returned to the web service caller:

- Response
- ID of workflow event
- State of workflow event
- Status of workflow event
- Action of the last executed workflow activity

When a Business Process Rule is set, the response is returned to the web service caller when the workflow is suspended and a work item is created.

**Figure 1  SYNCHR Execution Mode**

**Asynchronous Mode (ASYNCHR)**

With asynchronous services, the client invokes a service but does not wait for the completion of the workflow execution.

In TIBCO MDM, when a record is saved for an action, a workflow start event is generated and the event is sent to JMS for processing. In the Asynchronous mode, the response does not wait for the workflow execution to end. Before the workflow execution ends (that is, immediately after the event is sent to JMS for processing), the following is returned to the web service caller:

- Response
- Workflow event ID

By default, all web services which trigger a workflow are executed in the asynchronous mode.

This is the default execution mode.
Figure 2  ASYNCHR Execution Mode
Web Service Message Structure

The message structure of the web services is based on the following two types:

- Generic schema based
- Strict schema based

Generic Schema Based Message Structure

The generic schema based message structure is used for the following web services:

- Record Services on page 47
- Work Item Services on page 117
- Text Search Service on page 182

The following elements are present in a web service request or response:

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;DataService&gt;</code></td>
<td>The <code>DataService</code> element defines the boundary of a web service request. It is the outermost element in the XML. A message can contain one or more 'Transactions'.</td>
</tr>
<tr>
<td><code>&lt;Identity&gt;</code></td>
<td>The <code>Identity</code> element identifies a user or system making the web service call. The identity establishes the roles and permissions to be used while performing the requested operations. If the identity cannot be established from transport level headers (example: SOAP headers), the application uses this section to build a user profile. In scenarios, where a ‘session’ can be maintained with the server using cookies or other means, the ‘Identity’ section is not necessary for identification. In this case the server builds the profile using cookies.</td>
</tr>
<tr>
<td><code>&lt;Context&gt;</code></td>
<td>The <code>Context</code> element specifies a list of context variables that define the context of execution for the request. Some examples of context variables are <code>RECORD_ACTION</code>, <code>WORKITEMREFERENCE</code>, <code>CORRELATIONID</code>, and so on.</td>
</tr>
</tbody>
</table>
Common Sections

Some sections that are common across many Record and Work Item web services are:

---

**Table 2 Elements in a Web Service Request or Response**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Transaction&gt;</code></td>
<td>The <code>Transaction</code> element denotes the boundaries of a unit of work. All operations ('Commands') are executed under a common context. However, database transaction semantics, in particular bulk rollback, are not supported.</td>
</tr>
<tr>
<td><code>&lt;Command type=&quot;Query&quot;&gt;</code> <code>&lt;Command&gt;</code></td>
<td>The 'type' attribute of the <code>Command</code> element describes the type of operation being performed on an object. The allowed types, for example, are 'Query', 'Validate', 'Add' and 'Modify'. Each 'Command' can contain one or more 'Entities'. All entities included in a command have to be of the same type (example 'MasterCatalogRecord').</td>
</tr>
<tr>
<td><code>&lt;MasterCatalogRecord etype=&quot;Entity&quot; commandqualifier=&quot;ValidValues&quot;&gt;</code></td>
<td>Business objects in the application are modeled as Entities. Examples include 'MasterCatalogRecord' and 'WorkItem'.</td>
</tr>
<tr>
<td><code>&lt;Result severity=&quot;Error&quot; code=&quot;CAT-1292&quot;&gt;</code></td>
<td>The <code>&lt;Result&gt;</code> element describes what error occurred. It is aimed, primarily, at the human user who has to interpret why the message failed. Each <code>&lt;Result&gt;</code> element contains a severity attribute and a code. Severities indicate if this is an &quot;Error&quot;, &quot;Warning&quot; or &quot;Info&quot; message. The code is a TIBCO MDM internal code which can be used to track the error. The <code>&lt;ResultList&gt;</code> element can contain one or more <code>&lt;Result&gt;</code> entries. The <code>&lt;Description&gt;</code> element is a human readable description of the error.</td>
</tr>
<tr>
<td><code>&lt;TransactionResult result=&quot;Completed&quot;&gt;</code> <code>&lt;SuccessCount&gt;2&lt;/SuccessCount&gt;</code></td>
<td>The <code>&lt;TransactionResult&gt;</code> element captures the status of a transaction to let the caller know if the command succeeded or failed. A 'Success' here does not translate into the successful completion of the Command. However, a failure here does mean the Command was aborted. The <code>&lt;TransactionResult&gt;</code> element also captures details such as EventID, EventState, Action, and so on.</td>
</tr>
</tbody>
</table>
**ResultList**

The `<ResultList>` element captures the results of the web service. A ‘ResultList’ contains one or more errors, warnings or informational messages. The `<ResultList>` element can contain one or more `<Result>` entries. For example:

```xml
<ResultList highestSeverity="Error" reference="UOM">
  <Result severity="Error" code="CAT-1292">
    <Description lang="en">The UOM is not a valid value</Description>
  </Result>
  <Result severity="Error" code="CAT-1293">
    <Description lang="en">The UOM has to be a decimal</Description>
  </Result>
</ResultList>
```

In the above example, the attribute ‘UOM’ has 2 errors. Each ‘Result’ in the ‘ResultList’ has a ‘code’, ‘severity’ and ‘description’.

Code – is the error code associated with the error.

Severity – can be ‘Error’, ‘Info’ or ‘Warning’.

Description – provides the user a readable description of the error.

**TransactionResult**

The `<TransactionResult>` element captures the status of a transaction to let the caller know if the command succeeded or failed.

- SuccessCount — `SuccessCount` indicates the number of records successfully processed.
- TotalCount — `TotalCount` indicates the total number of records that met the search criteria of the request. This is useful since the user gets only a subset of the actual result set.

A ‘Success’ here does not translate into the successful completion of the command. However, a failure here does mean that the command was aborted. The `<TransactionResult>` element also captures details such as EventID, EventState, Action, and so on.

```xml
<TransactionResult result="Completed">
  <SuccessCount>1</SuccessCount>
  <TotalCount>1</TotalCount>
</TransactionResult>
```

The following is a sample of the TransactionResult when the execmode is set to SYNCHR.

- If the workflow is again suspended in another step, that is, another work item:

```xml
<TransactionResult result="Completed">
  <SuccessCount>1</SuccessCount>
</TransactionResult>
```
If the workflow completes:

```
<TransactionResult result="Completed">
  <SuccessCount>1</SuccessCount>
  <EventID>521859</EventID>
  <EventStatus>
    <Code>SUCCESS</Code>
    <Description lang="en">Success</Description>
  </EventStatus>
  <EventState>
    <Code>DONE</Code>
    <Description lang="en">Done</Description>
  </EventState>
  <Action>SetStatusToSuccess</Action>
</TransactionResult>
```

Comment History

The result also contains the comment history within the workflow, of which the work item is a part. The comment history is represented as an array of comments (‘marray’).

Work Item Locking

For more information, refer to Work Item Lock Service, on page 130.

Strict Schema Based Message Structure

This message structure is strictly schema compliant. You can easily comprehend it. The strict schema based message structure is used for the following web services:

- Administration Services on page 151
- Data Services on page 181 except Text Search Service on page 182

Note that the Text Search web service has a specific structure suited to searching records using search terms.
- Get Content Service on page 236
- Event Management Services on page 239

Samples

Sample request and response XML files are available in each folder specific to the web service:

$MQ_HOME/schema/DataService/2.0/samples
## Accessing Web Services

The URL the web service request is sent to is called an endpoint address. It allows the server to invoke the intended operation. The way in which a server finds the service and the operation to be invoked is called dispatching. TIBCO MDM supports two different ways to resolve a service and operation — Endpoint dispatching and SOAP Action dispatching. Releases prior to the current release only supported SOAP action dispatching.

- **Endpoint dispatching** — Append the operation name to the service URL. For example, in the service URL (http://<host>:<port>/eml/services/router), append the operation name (RouteAction). The endpoint URL should be http://<host>:<port>/eml/services/router/RouteAction.

- **SOAP action dispatching** — Use the endpoint URL of a service to address the service. Use the SOAP Action for identifying an operation.

This release of TIBCO MDM supports SOAP actions from version 2.0 of the schema.

### Endpoint Dispatching

<table>
<thead>
<tr>
<th>Service</th>
<th>Endpoint URL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Service</td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router/RouteAction</td>
<td>For requests wrapped in a Payload.</td>
</tr>
<tr>
<td></td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router/MasterCatalogRecordAction</td>
<td>For requests accessing master data.</td>
</tr>
<tr>
<td></td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router/WorkitemAction</td>
<td>For requests accessing work item functionality.</td>
</tr>
<tr>
<td>Text Search Service</td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/DataService/textSearchAction</td>
<td>For text search requests.</td>
</tr>
</tbody>
</table>
SOAP Action Dispatching

SOAP Action dispatching is mainly used for schema version 1.5.

Table 4  SOAP Action Dispatching

<table>
<thead>
<tr>
<th>Service</th>
<th>Endpoint URL</th>
<th>SOAP Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Service</td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router</td>
<td>RouteAction</td>
<td>For requests wrapped in Payload (current release).</td>
</tr>
<tr>
<td></td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router</td>
<td><a href="http://www.tibco.com/ci">http://www.tibco.com/ci</a> m/dataservice/2.0/wsd1/routeaction</td>
<td>For requests wrapped in Payload (releases prior to current release).</td>
</tr>
<tr>
<td></td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router</td>
<td>MasterCatalogRecordAction</td>
<td>For requests accessing master data – (current release).</td>
</tr>
<tr>
<td></td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router</td>
<td><a href="http://www.tibco.com/ci">http://www.tibco.com/ci</a> m/dataservice/2.0/wsd1/mastercatalogrecordaction</td>
<td>For requests accessing master data (releases prior to current release).</td>
</tr>
<tr>
<td></td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router</td>
<td>WorkitemAction</td>
<td>For requests accessing work item functionality (current release).</td>
</tr>
<tr>
<td></td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/router</td>
<td><a href="http://www.tibco.com/ci">http://www.tibco.com/ci</a> m/dataservice/2.0/wsd1/workitemaction</td>
<td>For requests accessing work item functionality (releases prior to current release).</td>
</tr>
<tr>
<td>Text Search Service</td>
<td>http://&lt;host&gt;:&lt;port&gt;/eml/services/DataService</td>
<td>textSearchAction</td>
<td>For text search requests (current release).</td>
</tr>
</tbody>
</table>
Validation of XML

By default, an incoming request XML is not validated against any XML schema (XSD file). However, you can enable the validation using the Configurator (Miscellaneous > Web Service Request Validation).

When validation is enabled, TIBCO MDM validates every request XML instance for the following:

- Namespace and Schema Reference — Ensures that the XML instance has used the correct namespace and schema reference.
- Inclusion of xsi:type — When EntityData elements are used, TIBCO MDM checks with the schema to see whether a specific type should be used by using xsi:type.

Validation adds performance overhead and needs to be analyzed before enabling it.

After the validation, TIBCO MDM executes the request if there are no validation errors. If validation is unsuccessful, validation errors are returned.

Validation of XML Instances

When validation is turned off (default):

- The response XML instances have namespace and schema reference.
- The output format of repository record attributes in the XML response of repository record actions add, modify and validate.

When validation is turned on:

- All the request XML instances contain namespace and schema reference.
- All the response XML instances contain namespace and schema reference.
- The responses for repository ‘Add’ and ‘Modify’ command types are returned in an attribute form and not the element form.

Namespace and Schema Reference

TIBCO MDM supports validation XML instances using namespaces. The namespaces allow for scoping of the XML elements so that name collisions do not occur when an application uses XML instances from multiple sources.
Inclusion of xsi:type

The XML schema design uses inheritance with extension model via xs:extension to add more elements or restrict the domain of values for schema components. Following this design, the EntityData element used in request and response XML documents must use the correct xsi:type in the XML request and response documents. The EntityData is a base type and the XML instance will contain specific types depending on the context. The implementation of web services is modified to include the xsi:type whenever a specific type needs to be present in the instance.

Service Methods

TIBCO MDM supports validation of request XML for the following service methods:

- MasterCatalogRecordAction
- RouteAction
- WorkItemAction

The schemas published for these web service methods use a specific namespace for each of the service method’s request and responses. The following namespaces and corresponding schemas are used for validation of various requests and responses.

The schema for web services is available at $MQ_HOME/schema/DataService/2.0.

For a list of other web services available, refer to Web Services Available in TIBCO MDM, page 39.

MasterCatalogRecordAction

This service method is used to invoke catalog record related functionality such as add, modify, validate, and so on. The request and response XML instances for this service method must be well-formed and compliant with their corresponding schemas:

- Request — MasterCatalogRecord_Action_Request.xsd
- Response — MasterCatalogRecord_Action_Response.xsd

RouteAction

This service method is used to invoke both catalog related functions and work item related functions. The argument to this service method is passed as a CDATA block under the payload element (Payload.xsd).

- Payload Request — Payload.xsd
Payload Response — Payload.xsd

Encapsulating payload in CDATA block offers slightly higher performance as the payload is not parsed by the transport layers. The contents of the payload type attribute indicate whether the request is for catalog record action or work item action. The response XML instance is also encapsulated in payload element.

Table 5  Attributes of Payload.xsd

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Determines the type of payload contents. The type can be either mastercatalog or workitem. The default type is mastercatalog. Introducing the type attribute makes it easier to identify a request as catalog-related or work-item related without &quot;looking&quot; into the contents.</td>
</tr>
</tbody>
</table>

WorkItemAction

This service method is used to invoke the work item related functions such as query work item, close work item, and so on. The request and response XML instances for this service method must be well-formed and compliant with their corresponding schemas:

- Work Item Request — Workitem_Action.xsd

  The work item request schema supports the capturing of arbitrary user inputs. It uses the FormResult element instead of the Accept element. Valid values are 'Accept' and 'Reject'. Any other value passed is ignored and defaulted to Accept.

  When validation is turned on, the arbitrary input from the user application must be passed within the CustomAttributes element as follows:

  ```xml
  <FormResult etype="Attribute" type="string">Accept</FormResult>
  <CustomAttributes>
      <raiseAlarm type="string">Alarm string</raiseAlarm>
      <raiseAlarm1 type="string">Alarm string</raiseAlarm1>
      <Decision etype="Attribute" type="string">val</Decision>
  </CustomAttributes>
  ```

  Since release 8.2.0 onwards, the FormResult parameter is deprecated, you are recommended to use Close/Approve, Close/Reject or CloseAll/Approve, or CloseAll/Reject.
The following is the schema component that supports arbitrary user inputs.

```xml
<xs:complexType name="WorkItemCustAttrsType" mixed="true">
  <xs:sequence>
    <xs:any namespace="##any" processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

- Work Item Response — `Workitem_Action.xsd`
Including Namespace in Response

You can control the addition of namespace in the response by setting the Include of namespace property in Configurator. The default is true, that is, the namespace is included in the response.

If the namespace is included, it can be customized by use of the Miscellaneous > Webservice schema version property in Configurator. For example, this release of TIBCO MDM contains the 2.0 version of the schema.
Design Principles

- All web service calls work on the request-response paradigm. To achieve a task, a user sends in a request XML, and the results of the request are sent back in a response XML.

- All web service calls are synchronous. A response is always sent, even if processing of a request is delegated to the workflow.

- Each web service request or call must be complete. Every time TIBCO MDM receives a request, it expects all data necessary (for it to handle the request) to be available in the request XML.

- No state is maintained between web service calls. TIBCO MDM does not enforce the order in which web service calls should be made. You can make any web service call anytime.

- ‘Query’ and ‘Validate’ messages do not change the state of the system. By definition, ‘Query’ messages are read only and do not change the state of the system. This means, you can call the ‘Query’ service on an entity any number of times without causing the state of the system to change.

- ‘Add’ and ‘Modify’ messages can change state of the system.

- ‘Add’ and ‘Modify’ messages respect transactions. If a web service fails, all activity pertaining to the request is rolled back. A web service request can contain multiple transactions, which are independent of each other. For example, if the first transaction is successfully executed and the second transaction fails, there is no effect of the second transaction on the first.
Authentication

Authentication for web services works as follows:

- If a web service request is received from a client without specifying any existing HTTP session the user is authenticated.

- If a web service request specifies an existing session and if the session is not already expired, the session is maintained and no authentication is done. In this case, no password needs to be specified.

Following authentication mechanisms are supported:

- Normal/Default mode: A password needs to be specified. The authentication is done using the user ID and password. User can reside in either database or LDAP. However, if a valid web service session ID is specified by appending ;jsessionid=<session id value> to the web service request URL, authentication is skipped provided the specified HTTP session is valid.

  If an HTTP session is present for an existing user, the user ID is also specified in the web service request, and if the user ID does not match with the session owner; user ID in the web service request takes precedence and the user is re-authenticated into TIBCO MDM. The HTTP session is invalidated later on.

- Single Sign-On (SSO) mode: Single sign-on is a mechanism whereby a single initial action of user authentication and authorization allows you to access subsequent multiple web services for which you are authorized, without the need to specify password for all such subsequent requests. All TIBCO MDM web services are SSO-enabled.

  TIBCO MDM supports variations of single sign-on. For example, Site Minder/Transaction Minder (SM) and SAML2.0. For the purpose of trust verification and validating user data, TIBCO MDM relies on the information or attributes set in the SOAP headers. This helps in establishing the trust and confirming that request has received from a reliable or trusted source and user is pre-authenticated. Role information for the user is also extracted from these headers. If user does not exist in TIBCO MDM, it is created. If the user data relevant for TIBCO MDM changes over subsequent web service requests, it gets updated in the application. TIBCO MDM does not store user password in this mode.
The authentication of the user name is controlled by the User name case sensitivity property that is specified in Configurator (InitialConfig > Miscellaneous). By default, the value is true.

Tivoli Access Manager (TAM) does not support web services single sign-on. However, it can be customized to secure web service requests, and TIBCO MDM can work with that through the same model of establishing trust through header information. TIBCO MDM does not support TAM based single sign-on out of the box.

**Using Secure Socket Layer (SSL) with Web Services**

TIBCO MDM uses AXIS2 as a provider of web services. To serve the web service requests over https; AXIS2 needs to be configured to receive messages on HTTPS transport. Hence, the endpoint reference or URLs starting with https:// can be served. This requires configuring transportReceiver for https in axis2.xml.

The following section needs to be uncommented and configured in axis2.xml. This file is provided in $MQ_HOME/ECM.ear/EML.war/WEB-INF/conf directory.

```xml
<!-- the non blocking https transport based on HttpCore + SSL-NIO extensions
<transportReceiver name="https" class="org.apache.axis2.transport.nhttp.HttpCoreNIOSSLListener">
  <parameter name="port" locked="false">9002</parameter>
  <parameter name="non-blocking" locked="false">true</parameter>
  <parameter name="keystore" locked="false">true</parameter>
    <KeyStore>
      <Location>identity.jks</Location>
      <Type>JKS</Type>
      <Password>password</Password>
      <KeyPassword>password</KeyPassword>
    </KeyStore>
  </parameter>
  <parameter name="truststore" locked="false">
    <TrustStore>
      <Location>trust.jks</Location>
      <Type>JKS</Type>
      <Password>password</Password>
    </TrustStore>
  </parameter>
  <!---<parameter name="SSLVerifyClient">require</parameter> supports optional|require or defaults to none -->
```
This does not support client certification authentication, that is, it is one way SSL.

The new web services such as Administrator, Data, Content, and Event Management web services require various types of authentication. The subsequent sections provide the detailed information on authentication for each of these web services.

- Administrator Web Services, page 20
- Data Web Services, page 22
- DataLoad Web Services, page 23
- Content Web Services, page 23
- Event Management Web Services, page 25

### Administrator Web Services

The following table lists authentication required for Administrator web services:

#### Table 6  Authentication of Administrator Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Menu Privilege</th>
<th>Function Privilege</th>
<th>Role Privilege</th>
<th>Resource Access Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login</td>
<td>NA</td>
<td>NA</td>
<td>Any</td>
<td>NA</td>
</tr>
<tr>
<td>Logout</td>
<td>NA</td>
<td>NA</td>
<td>Any</td>
<td>NA</td>
</tr>
<tr>
<td>Get Entitlement</td>
<td>NA</td>
<td>Allow view entitlement</td>
<td>Administrator</td>
<td>NA</td>
</tr>
<tr>
<td>Create Enterprise</td>
<td>Administration &gt; All Company Profiles</td>
<td>adminmenu_company</td>
<td>Super User</td>
<td>Create Company Profile</td>
</tr>
<tr>
<td>Get Enterprise List</td>
<td>Administrator &gt; All Company Profiles</td>
<td>adminmenu_company</td>
<td>Super User, Administrator, and Work Supervisor</td>
<td>Get All Company Profiles</td>
</tr>
<tr>
<td>Create User</td>
<td>Administration &gt; User Accounts</td>
<td>adminmenu_member</td>
<td>Super User, Administrator, and Work Supervisor</td>
<td>Create User</td>
</tr>
</tbody>
</table>
Table 6  Authentication of Administrator Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Menu Privilege</th>
<th>Function Privilege</th>
<th>Role Privilege</th>
<th>Resource Access Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modify User</td>
<td>Administration &gt; User Accounts</td>
<td>adminmenu_member</td>
<td>Super User, Administrator, and Work Supervisor</td>
<td>Modify user</td>
</tr>
<tr>
<td>Change Password</td>
<td>Administration &gt; User Accounts</td>
<td>adminmenu_member</td>
<td>Super User, Administrator, and Work Supervisor</td>
<td>Change Password</td>
</tr>
<tr>
<td>Delete User</td>
<td>Administration &gt; User Accounts</td>
<td>adminmenu_member</td>
<td>Super User, Administrator, and Work Supervisor</td>
<td>Delete user</td>
</tr>
<tr>
<td>Get User List</td>
<td>Administration &gt; User Accounts</td>
<td>adminmenu_member</td>
<td>Super User, Administrator, and Work Supervisor</td>
<td>Get User</td>
</tr>
<tr>
<td>Get Role List</td>
<td>Administration &gt; Roles</td>
<td>adminmenu_roles</td>
<td>Super User, Administrator, and Work Supervisor</td>
<td>Get Role</td>
</tr>
</tbody>
</table>
## Data Web Services

The following table lists authentication required for Data web services:

**Table 7  Authentication of Data Web Services**

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Menu Privilege</th>
<th>Function Privilege</th>
<th>Role Privilege</th>
<th>Resource Access Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text Search</td>
<td>Browse and Search</td>
<td>catalogmenu_catalog</td>
<td>Super User, Administrator, Work Supervisor, Repository Manager, Repository Editor, Repository Approver and Data Steward</td>
<td>Text Search</td>
</tr>
<tr>
<td>Get DataSource List</td>
<td>Master Data &gt; Data Sources</td>
<td>catalogmenu_data_source</td>
<td>Any</td>
<td>NA</td>
</tr>
<tr>
<td>Get Repository List</td>
<td>Master Data &gt; Repositories</td>
<td>catalogmenu_master_catalog</td>
<td>Super User, Administrator, Work Supervisor, Repository Manager, and Data Steward</td>
<td>List</td>
</tr>
<tr>
<td>Initiate Synchronization</td>
<td>Master Data &gt; Synchronization profiles</td>
<td>catalogmenu_catalog</td>
<td>Super User, Administrator, Work Supervisor, Repository Manager, Data Steward</td>
<td>Synchronize Records</td>
</tr>
<tr>
<td>Upload Metadata</td>
<td>System Operations</td>
<td>sysopmenu</td>
<td>Super User and Administrator</td>
<td>NA</td>
</tr>
<tr>
<td>Extract Data</td>
<td>NA</td>
<td>NA</td>
<td>Super User and Administrator</td>
<td>NA</td>
</tr>
</tbody>
</table>
DataLoad Web Services

The following table lists authentication required for DataLoad web services:

Table 8  Authentication of DataLoad Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Menu Privilege</th>
<th>Function Privilege</th>
<th>Role Privilege</th>
<th>Resource Access Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get DataSource Info</td>
<td>Master Data &gt; Data Sources</td>
<td>catalogmenu_ data_source</td>
<td>Any</td>
<td>NA</td>
</tr>
<tr>
<td>Upload DataSource</td>
<td>Master Data &gt; Data Sources</td>
<td>catalogmenu_ data_source</td>
<td>Any</td>
<td>NA</td>
</tr>
<tr>
<td>ImportRecords</td>
<td>Master Data &gt; Repositories &gt; Manage Input Maps</td>
<td>catalogmenu_ master_catalog</td>
<td>Super User, Administrator, Work Supervisor, Repository Manager, Data Steward, and Support Engineer</td>
<td>Import Records</td>
</tr>
<tr>
<td>LoadImportRecords</td>
<td>Master Data &gt; Repositories &gt; Manage Input Map</td>
<td>catalogmenu_ master_catalog</td>
<td>Super User, Administrator, Work Supervisor, Repository Manager, Data Steward, and Support Engineer</td>
<td>Import Records</td>
</tr>
</tbody>
</table>
## Table 9  Authentication of Content Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Menu Privilege</th>
<th>Function Privilege</th>
<th>Role Privilege</th>
<th>Resource Access Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Content</td>
<td>Event Log</td>
<td>adminmenu_event_log</td>
<td>Administrator, Super User</td>
<td>Event -&gt; Resource Access All</td>
</tr>
</tbody>
</table>
Event Management Web Services

The following table lists authentication required for Event Management web services:

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Menu Privilege</th>
<th>Function Privilege</th>
<th>Role Privilege</th>
<th>Resource Access Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Event Info</td>
<td>Event Log</td>
<td>adminmenu_event_log</td>
<td>Administrator, Super User</td>
<td>Event -&gt; Resource Access All</td>
</tr>
<tr>
<td>Get Event Details</td>
<td>Event Log</td>
<td>adminmenu_event_log</td>
<td>Administrator, Super User</td>
<td>Event -&gt; Resource Access All</td>
</tr>
<tr>
<td>Get Import Event Status</td>
<td>Event Log</td>
<td>adminmenu_event_log</td>
<td>adminmenu_event_log</td>
<td>Event -&gt; Resource Access All</td>
</tr>
</tbody>
</table>
Faults

If you do not specify an adequate information in the request, the web services return various fault messages. The following sections list the generic fault messages that are returned in the response. However, for detailed information on various types of error messages, refer to Appendix E - Error Codes in the TIBCO MDM System Administrator’s Guide.

- Administrator Web Services, page 26
- Data Web Services, page 28
- DataLoad Web Services, page 29
- Content Web Services, page 31
- Event Management Web Services, page 32

Administrator Web Services

The following table lists the faults of Administrator web services:

Table 11 Faults of Administrator Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login</td>
<td>If the specified credentials are wrong, Authentication failed for user &lt;Parameter name='USER'&gt; and company &lt;Parameter name='ENTERPRISE'&gt; message is returned.</td>
</tr>
<tr>
<td>Logout</td>
<td>If you get any errors while logging out, Error during logout. Invalid or expired session message is returned.</td>
</tr>
<tr>
<td>Get Entitlement</td>
<td>If user does not have privilege, User '&lt;user name&gt;' does not have permission to view entitlement message is returned.</td>
</tr>
<tr>
<td>Create Enterprise</td>
<td>If user does not have privilege, User &lt;USERNAME&gt; not authorized to Create Enterprise message is returned.</td>
</tr>
<tr>
<td>Get Enterprise List</td>
<td>If other than Administrator and Super User invokes this web service, the authentication failure message is returned.</td>
</tr>
<tr>
<td>Create User</td>
<td>If user does not have privilege, User &lt;USERNAME&gt; not authorized to Create User message is returned.</td>
</tr>
</tbody>
</table>
Modify User

If you specify wrong values for the attributes in the `request.xml` file, an error message is returned. For example,

- Wrong parameters specified in the `<UserInfo>` tag, `Invalid company <companyname>` message is returned.
- User credentials specified in the `<UserInfo>` tag other than administrator and work supervisor, `User <username> not authorized to Modify User` message is returned.
- If the specified user name does not exist, `User not defined` message is returned.
- If wrong enterprise name is specified in the `<ModifyUserDetails>` tag, the `Cannot find specified company <companyname>` message is returned.
- Mismatch combination in the `<LocaleSettings>` tag, the `Locale <languagename> is not supported` message is returned.
- If username is not specified, `Enterprise cannot be specified if user is not specified` message is returned.
- If first name is not specified, `First Name mandatory. Provide First Name` message is returned.
- If first name is modified more than 80 characters, `Length of First Name more than maximum allowed length 80` message is returned.
- If last name is modified more than 80 characters, `Length of Last Name more than maximum allowed length 80` message is returned.
- If middle name is modified more than 80 characters, `Length of MiddleName more than maximum allowed length 80` message is returned.
- If password is modified more than 30 characters, `Length of Password more than maximum allowed length 30` message is returned.

Change Password

If you do not specify a password in the `request.xml` file, the `Password should not be null or blank` message is returned.

Delete User

If user does not have privilege, `User <USERNAME> not authorized to Delete User` message is returned.

Table 11  
Faults of Administrator Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modify User</td>
<td>If you specify wrong values for the attributes in the <code>request.xml</code> file, an error message is returned. For example,</td>
</tr>
<tr>
<td></td>
<td>- Wrong parameters specified in the <code>&lt;UserInfo&gt;</code> tag, <code>Invalid company &lt;companyname&gt;</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- User credentials specified in the <code>&lt;UserInfo&gt;</code> tag other than administrator and work supervisor, <code>User &lt;username&gt; not authorized to Modify User</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If the specified user name does not exist, <code>User not defined</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If wrong enterprise name is specified in the <code>&lt;ModifyUserDetails&gt;</code> tag, the <code>Cannot find specified company &lt;companyname&gt;</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- Mismatch combination in the <code>&lt;LocaleSettings&gt;</code> tag, the <code>Locale &lt;languagename&gt; is not supported</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If username is not specified, <code>Enterprise cannot be specified if user is not specified</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If first name is not specified, <code>First Name mandatory. Provide First Name</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If first name is modified more than 80 characters, <code>Length of First Name more than maximum allowed length 80</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If last name is modified more than 80 characters, <code>Length of Last Name more than maximum allowed length 80</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If middle name is modified more than 80 characters, <code>Length of MiddleName more than maximum allowed length 80</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td>- If password is modified more than 30 characters, <code>Length of Password more than maximum allowed length 30</code> message is returned.</td>
</tr>
<tr>
<td>Change Password</td>
<td>If you do not specify a password in the <code>request.xml</code> file, the <code>Password should not be null or blank</code> message is returned.</td>
</tr>
<tr>
<td>Delete User</td>
<td>If user does not have privilege, <code>User &lt;USERNAME&gt; not authorized to Delete User</code> message is returned.</td>
</tr>
</tbody>
</table>
### Table 11  Faults of Administrator Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get User List</td>
<td>If the user list is not available, <strong>No Users exist</strong> message is returned.</td>
</tr>
<tr>
<td>Get Role List</td>
<td>If the role list is not available, <strong>No Roles exist</strong> message is returned.</td>
</tr>
</tbody>
</table>

### Data Web Services

The following table lists the faults of Data web services:

### Table 12  Faults of Data Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text Search</td>
<td>Various validation messages are returned. For more information, refer to the Appendix E - Error Codes in the <em>TIBCO MDM System Administrator’s Guide</em>.</td>
</tr>
<tr>
<td>Get DataSource List</td>
<td>If the data source does not exist, the <strong>CimFaultException</strong> error is returned.</td>
</tr>
<tr>
<td>Get Repository List</td>
<td>If the repository list is not available in the specified enterprise, <strong>No Repository Found</strong> message is returned.</td>
</tr>
<tr>
<td>Initiate Synchronization</td>
<td>If the Synchronization profile or the repository does not exist, <strong>Repository or output map with name or ID &lt;Parameter name='NAME'&gt; does not exist; may have been already deleted by another user</strong> message is returned in the response.</td>
</tr>
<tr>
<td>Upload Metadata</td>
<td>If user does not have privilege, <strong>User &lt;USERNAME&gt; not authorized to Upload Metadata</strong> message is returned.</td>
</tr>
<tr>
<td>Extract Data</td>
<td>If the class name is not provided, <strong>&lt;IDataProvider/IDataProcessor&gt; class name is mandatory. Please provide class name</strong> message is returned.</td>
</tr>
</tbody>
</table>
The following table lists the faults of DataLoad web services:

Table 13  Faults of DataLoad Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get DataSource Info</td>
<td>If data source name is not specified, the <strong>Data source name is mandatory. Provide the data source name</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If the data source does not exist or invalid, <strong>Invalid data source name</strong> error is returned.</td>
</tr>
</tbody>
</table>
### Faults of DataLoad Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upload DataSource</td>
<td>If the specified data source name is incorrect, <strong>Invalid data source name</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If the specified enterprise name is incorrect, <strong>Invalid company &lt;company_name&gt;</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If the specified password is incorrect, <strong>Cannot authenticate credentials with user name user_name, domain domain_name</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If the specified user name is incorrect, <strong>User user_name does not exist</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you do not specify any value for the <code>&lt;DatasourceName&gt;</code> parameter, <strong>Data source name is mandatory. Provide the data source name</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you do not specify any value for the <code>&lt;FileName&gt;</code> parameter, <strong>Data source format is not SQL. Filename is mandatory</strong> error is returned.</td>
</tr>
<tr>
<td>ImportRecords</td>
<td>If you select DatabaseLoader as the Import approval option with Synchronous execution mode, <strong>DBLoader doesn’t support for synch execution mode</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you do not specify any value for the <code>execmode</code> attribute, <strong>Invalid execution mode excutionmodename. Supported execution modes: ‘ASYNCHR’ or ‘SYNCHR’</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you specify non-existing value for the <code>InputMapName</code> parameter, <strong>Invalid parameter inputmapname specified</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you specify non-existing value for the <code>Repository</code> parameter, <strong>Repository or output map with name or ID repositoryname does not exist; may have been already deleted by another user</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If the specified enterprise name is incorrect, <strong>Invalid company &lt;company_name&gt;</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If the specified user name is incorrect, <strong>User user_name does not exist</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If the specified password is incorrect, <strong>Cannot authenticate credentials with user name user_name, domain domain_name</strong> error is returned.</td>
</tr>
</tbody>
</table>
### Table 13  Faults of DataLoad Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadImportRecords</td>
<td>If you do not specify any value for the Repository parameter, <strong>Repository name is mandatory, provide repository name</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you do not specify any value for the InputMapName parameter, <strong>Input map is mandatory, provide input map name</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you specify non-existing value for the InputMapName parameter, <strong>Invalid parameter inputmapname specified</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you specify non-existing value for the Repository parameter, <strong>Repository or output map with name or ID repositoryname does not exist; may have been already deleted by another user</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you upload an invalid file, the <strong>File Format not supported</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you do not specify any value for the &lt;BinaryData&gt; parameter, <strong>Cannot upload specified file 'VALUE'; invalid file name or file does not exist or is empty</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you do not specify any value for the &lt;FileName&gt; parameter, <strong>Data source format is not SQL. Filename is mandatory</strong> error is returned.</td>
</tr>
<tr>
<td></td>
<td>If you select DatabaseLoader as the Import approval option with the Synchronous execution mode, <strong>DBLoader doesn’t support for synch execution mode</strong> error is returned.</td>
</tr>
</tbody>
</table>

### Content Web Services

The following table lists the faults of Content web services:

### Table 14  Faults of Content Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Content</td>
<td>If the specified DocumentId or DocumentPath for a file does not exist or the user does not have &quot;View Event&quot; permissions, <strong>Could not find document with primary key id &lt;id&gt; or Not authorized to view event details</strong> message is returned.</td>
</tr>
</tbody>
</table>
Event Management Web Services

The following table lists the faults of Event Management web services:

Table 15  Faults of Event Management Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Menu Privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Event Info</td>
<td>The following messages are returned in response:</td>
</tr>
<tr>
<td></td>
<td>• Not authorized to view event details.</td>
</tr>
<tr>
<td></td>
<td>• Event not yet started.</td>
</tr>
<tr>
<td></td>
<td>• Event does not exist or executed in memory; no other details available.</td>
</tr>
<tr>
<td>Get Event Details</td>
<td>If the event ID is not found, Event does not exist or executed in memory; no other details available message is returned.</td>
</tr>
<tr>
<td>Get Import Event Status</td>
<td>The following messages are returned in response:</td>
</tr>
<tr>
<td></td>
<td>• If the specified enterprise name is incorrect, Invalid company &lt;company_name&gt; error is returned.</td>
</tr>
<tr>
<td></td>
<td>• If the specified password is incorrect, Cannot authenticate credentials with user name &lt;user_name&gt;, domain &lt;domain_name&gt; error is returned.</td>
</tr>
<tr>
<td></td>
<td>• If the specified user name is incorrect, User user_name does not exist error is returned.</td>
</tr>
<tr>
<td></td>
<td>• If an Event ID is incorrect then Event does not exist or executed in memory; no other details available error is returned.</td>
</tr>
</tbody>
</table>
Password Encryption and Decryption

The Password input is required across all web services and majorly in Administrator web services. You can encrypt any text password using the passwordEncryptDecryptTool.sh or passwordEncryptDecryptTool.bat tool located in $MQ_HOME/bin.

The scripts accept a text string and return an encrypted string value or accept an already encrypted string value and return a text value. You can generate an encrypted string using the tool and then use it in the web service request.

Prerequisite

Ensure that all environment variables used in TIBCO MDM are set properly before executing the script, such as MQ_HOME, MQ_COMMON_DIR, MQ_CONFIG_FILE, NODE_ID, WAS_HOME, JBOSS_HOME, and so on.

To encrypt the password:

1. Go to the command prompt; go to the $MQ_HOME/bin folder.
2. Type passwordEncryptDecryptTool.bat "<password>" and press ENTER. For example,

D:\work\venus_dest\bin>passwordEncryptDecryptTool.bat "tibco123"

D:\work\venus_dest\bin>Encoded tibco123: 4dc8a3f3d1e43f

Using Cipher Algorithm: PBEWithMD5AndDES

log4j:WARN No appenders could be found for logger (com.tibco.mdm.infrastructure.encryption.EncryptorDecryptorUtil).
log4j:WARN Please initialize the log4j system properly.

Supplied string:tibco123, Encrypted string:4dc8a3f3d1e43f

D:\work\venus_dest\bin>
The utility provides information about the accessed property file and loaded properties, and then displays the encrypted password.

For example, the encrypted password for tibco123 is cfMrxokzKTm3zJJLpDW7Jg==. 

3. Place the encrypted password in <ns:Password></ns:Password> tag of the web service request. For example, 

   <ns:Password>cfMrxokzKTm3zJJLpDW7Jg==</ns:Password>

To decrypt the password:

Use the similar encryption procedure to decrypt the password. Place the encrypted string in the command prompt instead of the text password. For example, $MQ_HOME/bin>passwordEncryptDecryptTool.bat "cfMrxokzKTm3zJJLpDW7Jg==". After entering the encrypted string as input, the utility returns the original decrypted password. In this case, tibco123.

You must specify double quotes for the text password or encrypted password, else the utility does not perform encryption or decryption.

Encryption and decryption tool is also supported through a Java API so that it can be embedded.

Usage is as follows:

```java
import com.tibco.mdm.infrastructure.encryption.EncrypterDecryptorUtil;
...
...
EncrypterDecryptorUtil cipher = EncrypterDecryptorUtil.getCipher();
cipher.encrypt(<pwdStringInPlainText>);
OR
cipger.decrypt(<EncryptedPwdValue>);
```
Session Handling

TIBCO MDM web services allow the SOAP client (web client, BW-client) to maintain a session between web service calls, so the client need not provide the user credentials in subsequent requests called.

The following table illustrates various scenarios involved in a web service session management behavior in TIBCO MDM.

**Table 16  UI Browser Web Client**

<table>
<thead>
<tr>
<th>WS request launched within running TIBCO MDM</th>
<th>URL Writing</th>
<th>Cookie enable</th>
<th>User Information Provided</th>
<th>Session exists</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y (Existing session is used, that is, the one created by the TIBCO MDM UI) Username from 'userInfo' section of the web service request is checked with profile name. No new session is created.</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>No new session is created. The existing session is used.</td>
</tr>
<tr>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y (Existing session is used, that is, the one created by the TIBCO MDM UI) No new session is created, but the web service session ID needs to be passed to the GI based web service client which appends the jsessionid=&lt;&gt; to the endpoint URL. <strong>Note:</strong> Currently, the passing of sessionId to GI Application is not supported.</td>
</tr>
</tbody>
</table>
### Table 16  UI Browser Web Client

<table>
<thead>
<tr>
<th>WS request launched within running TIBCO MDM</th>
<th>URL Writing</th>
<th>Cookie enable</th>
<th>User Information Provided</th>
<th>Session exists</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N (Launched in a different browser window)</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y (Existing session is used, that is, the one created by the TIBCO MDM UI)</td>
<td>No new session is created; existing session is used. <strong>Note:</strong> The login service should be invoked first.</td>
</tr>
<tr>
<td>N (Launched in a different browser window)</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y (Existing session created using the Login service is used)</td>
<td>Username from 'userInfo' section of the web service request is checked with profile name. No new session is created.</td>
</tr>
<tr>
<td>N (Launched in a different browser window)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y (Existing session created using the Login service is used)</td>
<td>No new session is created, but as cookies are not enabled jsessionid=&lt;&gt; needs to be appended to the endpoint URL.</td>
</tr>
<tr>
<td>N (Launched in a different browser window)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Throws an exception stating user does not exists.</td>
</tr>
<tr>
<td>N (Launched in a different browser window)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>The invocation is stateless, that is, no new session is created if any service apart from Login service is invoked, even if the session does not exists.</td>
</tr>
</tbody>
</table>
TIBCO ActiveMatrix BusinessWorks

This section describes the case of a SOAP client based on TIBCO ActiveMatrix Businessworks.

The user wants to maintain a session (Statefull) across multiple web service calls. In this case, the user must invoke the Login service and BusinessWorks needs to maintain the sessionId returned from Login service in global context and append the jsessionid=<> to the endpoint URL.

- Ensure that endpoint URL contains the appropriate SOAP Action. For example,
  - For Login:
    http://<servername>:<port>/eml/services/AdminService/loginAction
  - For Add Record:
    http://<servername>:<port>/eml/services/router/MasterCatalogRecordAction;jsessionid=T2yJKLDpSY4CjcbzpG

- For WAS, ensure URL re-writing is enabled and the jsession ID is prefixed with 0000. For example, <0000jsessionID> is appended to the endpoint URL. For example,
  - For Add Record:
    http://<servername>:<port>/eml/services/router/MasterCatalogRecordAction;jsessionid=0000T2yJKLDpSY4CjcbzpG
  - For Logout:
    http://<servername>:<port>/eml/services/AdminService/logoutAction;jsessionid=0000T2yJKLDpSY4CjcbzpG

Prefixing 0000 with jsession ID is required because jsession ID cookie in WAS contains cache ID, which is prefixed to the session ID. The retrieved session ID in the response omits this cache ID, which is 0000 in most cases. Hence, you must explicitly prefix 0000 to the session ID while using a SOAP client.
The user credentials are provided at the time of login and for every subsequent call the jsessionid token is passed. The BW-client provides an input area for the SOAPRequestReply activity as shown in the following figure:

If the user does not want to maintain a session (Stateless), the user need not invoke the Login service. However, you need to provide the userInfo in every consecutive invocation of any service. If the userInfo is not provided, an error is displayed.
Web Services Available in TIBCO MDM

In TIBCO MDM, web services are available for the following activities:

<table>
<thead>
<tr>
<th>Web Service Type</th>
<th>Service</th>
<th>Description</th>
<th>Details Available on Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Record Services</td>
<td>Record Query Service</td>
<td>This web service can be used to query one or more repository records.</td>
<td>page 44</td>
</tr>
<tr>
<td></td>
<td>Record Query MetaData</td>
<td>This web service can be used for obtaining repository record meta-data.</td>
<td>page 44</td>
</tr>
<tr>
<td></td>
<td>ValidValues (Init)</td>
<td>This web service provides a list of:</td>
<td>page 88</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• initial and valid values for all attributes as defined by the rulebase</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>specified for the repository.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• access privileges configured for attributes.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• valid relationship types available for a given record.</td>
<td></td>
</tr>
<tr>
<td>Record Validate Service</td>
<td>Record Validate Service</td>
<td>This web service is used to validate the attribute values of a record</td>
<td>page 94</td>
</tr>
<tr>
<td></td>
<td></td>
<td>against the validation rules defined in the system.</td>
<td></td>
</tr>
<tr>
<td>Record Add Modify</td>
<td>Record Add Modify Service</td>
<td>This web service is used to add or modify a repository record. This service</td>
<td>page 96</td>
</tr>
<tr>
<td></td>
<td></td>
<td>works on record bundles, and can therefore take a list of records for</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>processing.</td>
<td></td>
</tr>
</tbody>
</table>
**Table 17  Web Services Available in TIBCO MDM**

<table>
<thead>
<tr>
<th>Web Service Type</th>
<th>Service</th>
<th>Description</th>
<th>Details Available on Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delete Record Service</td>
<td>This web service is used to delete:</td>
<td>page 105</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• a repository record</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• related target records</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• both the repository record and related target records.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• a record or relationship targets with reference to a work item</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Initiate Workflow Service</td>
<td>This web service is used to initiate a user-defined workflow for given record.</td>
<td>page 109</td>
</tr>
</tbody>
</table>
### Table 17  Web Services Available in TIBCO MDM

<table>
<thead>
<tr>
<th>Web Service Type</th>
<th>Service</th>
<th>Description</th>
<th>Details Available on Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work Item Services</td>
<td>Work item Query Parameters Service</td>
<td>This web service can be used to get a list of query parameters supported by the system and also provides a list of valid values for each of the supported parameters.</td>
<td>page 118</td>
</tr>
<tr>
<td></td>
<td>Work item Query Service</td>
<td>This web service can be used to get a list of query work items, based on the list of query parameters specified.</td>
<td>page 120</td>
</tr>
<tr>
<td></td>
<td>Work item Modify Close Service</td>
<td>This web service can be used to close a list of work items.</td>
<td>page 126</td>
</tr>
<tr>
<td></td>
<td>Work item Lock Service</td>
<td>This web service can be used to lock or relock a work item.</td>
<td>page 130</td>
</tr>
<tr>
<td></td>
<td>Work item UnLock Service</td>
<td>This web service can be used to unlock a work item.</td>
<td>page 132</td>
</tr>
<tr>
<td></td>
<td>Get Work item Service</td>
<td>This web service can be used to get the following details of a work item:</td>
<td>page 134</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Work item data</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Work item lock details</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Comment history</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Primary and associated records. For each record, operational data is also included.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Work item Reassign Service</td>
<td>This web service can be used to reassign a work item to another user.</td>
<td>page 140</td>
</tr>
</tbody>
</table>
### Table 17  Web Services Available in TIBCO MDM

<table>
<thead>
<tr>
<th>Web Service Type</th>
<th>Service</th>
<th>Description</th>
<th>Details Available on Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator Services</td>
<td>Login Service</td>
<td>This web service allows you to login to TIBCO MDM after authenticating the specified credentials.</td>
<td>page 152</td>
</tr>
<tr>
<td></td>
<td>Logout Service</td>
<td>This web service allows you to logout from the user session for the maintained session. The session is maintained either by URL re-writing mechanism or through cookies.</td>
<td>page 153</td>
</tr>
<tr>
<td></td>
<td>Get Entitlement Service</td>
<td>This web service allows you to retrieve entitlement information for a specified user. Entitlement information includes user permissions, roles assigned, and functions assigned to the user.</td>
<td>page 154</td>
</tr>
<tr>
<td></td>
<td>Create Enterprise Service</td>
<td>This web service allows you to create an enterprise in TIBCO MDM.</td>
<td>page 157</td>
</tr>
<tr>
<td></td>
<td>Create User Service</td>
<td>This web service allows you to create a user in an existing Enterprise.</td>
<td>page 161</td>
</tr>
<tr>
<td></td>
<td>Delete User Service</td>
<td>This web service allows you to delete a user in an existing Enterprise.</td>
<td>page 174</td>
</tr>
<tr>
<td></td>
<td>Get User List Service</td>
<td>This web service allows you to get a list of the currently logged-in users for an enterprise.</td>
<td>page 176</td>
</tr>
<tr>
<td></td>
<td>Get Role List Service</td>
<td>This web service allows you to get a list of roles of the currently logged-in users for an enterprise.</td>
<td>page 178</td>
</tr>
</tbody>
</table>
### Table 17  Web Services Available in TIBCO MDM

<table>
<thead>
<tr>
<th>Web Service Type</th>
<th>Service</th>
<th>Description</th>
<th>Details Available on Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Services</td>
<td>Text Search Service</td>
<td>The Text Search web service allows you to search for human recognizable terms in one or more TIBCO MDM repositories as you would do on an Internet Search Engine.</td>
<td>page 182</td>
</tr>
<tr>
<td></td>
<td>Get Repository List Service</td>
<td>This service allows you to get a list of repositories that can be accessed by the currently logged-in user.</td>
<td>page 193</td>
</tr>
<tr>
<td></td>
<td>Initiate Synchronization Service</td>
<td>This service allows you to synchronize records on a Trading Partner. This service operates in two modes, Async and Sync.</td>
<td>page 199</td>
</tr>
<tr>
<td></td>
<td>Upload Metadata Service</td>
<td>This service allows you to upload metadata (jar or xml file) to TIBCO MDM.</td>
<td>page 201</td>
</tr>
<tr>
<td></td>
<td>Extract Data Service</td>
<td>This service allows you to extract data from TIBCO MDM using custom implementation.</td>
<td>page 203</td>
</tr>
<tr>
<td>Content Services</td>
<td>Get Content Service</td>
<td>This service allows you to download a file for the specified context. This service takes DocumentId or DocumentPath as input, also the context from where the file needs to download is specified. If the context is accessible for the user only then the file is downloaded.</td>
<td>page 236</td>
</tr>
<tr>
<td>Event Management Services</td>
<td>Get Event Service</td>
<td>This service allows you to get event information for a specified event ID.</td>
<td>page 240</td>
</tr>
<tr>
<td></td>
<td>Get Event Details Service</td>
<td>This service allows you to retrieve the event details for the specified Event ID.</td>
<td>page 242</td>
</tr>
</tbody>
</table>
Web Services Accepting Multiple Entities

A TIBCO MDM web service requests top-level business objects as Entities. MasterCatalogRecord and WorkItem are the examples of an entity. These examples represent in a web service as `<MasterCatalogRecord etype="Entity">` or `<Workitem etype="Entity">` in the `<Command>` element.

Most of the web services can work only on one unique business object or an entity. However, some web services can work on more than one such business object or an entity as a part of a single web service request.

The following table lists if a web service supports multiple business objects or entities.

Table 18  Web Services Accepting Multiple Entities

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Does it accept multiple entities?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Record Services</strong></td>
<td></td>
</tr>
<tr>
<td>Record Query Service</td>
<td>No</td>
</tr>
<tr>
<td>Record Query MetaData Service</td>
<td>No</td>
</tr>
<tr>
<td>Record Query ValidValues (Init) Service</td>
<td>No</td>
</tr>
<tr>
<td>Record Validate Service</td>
<td>Yes</td>
</tr>
<tr>
<td>Record-Add Modify Services</td>
<td>Yes (with two transactions in one request)</td>
</tr>
<tr>
<td>Delete Record Service</td>
<td>Yes</td>
</tr>
<tr>
<td>Initiate Workflow Service</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Work Item Services</strong></td>
<td></td>
</tr>
<tr>
<td>Work Item Query Parameters Service</td>
<td>No</td>
</tr>
<tr>
<td>Work Item Query Service</td>
<td>No</td>
</tr>
<tr>
<td>Work Item Modify Close Service</td>
<td>Yes</td>
</tr>
<tr>
<td>Work Item Lock Service</td>
<td>Yes</td>
</tr>
<tr>
<td>Work Item UnLock Service</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### Web Services Accepting Multiple Entities

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Does it accept multiple entities?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work Item Get Service</td>
<td>Yes</td>
</tr>
<tr>
<td>Work Item Reassign Service</td>
<td>Yes (with two transactions in one request)</td>
</tr>
<tr>
<td>Get Custom Work Item Summary Service</td>
<td>No</td>
</tr>
<tr>
<td>Get Predefined Work Item Summary Service</td>
<td>No</td>
</tr>
<tr>
<td><strong>Administration Services</strong></td>
<td></td>
</tr>
<tr>
<td>Login Service</td>
<td>No</td>
</tr>
<tr>
<td>Logout Service</td>
<td>No</td>
</tr>
<tr>
<td>Get Entitlement Service</td>
<td>No</td>
</tr>
<tr>
<td>Create Enterprise Service</td>
<td>No</td>
</tr>
<tr>
<td>Get Enterprise List Service</td>
<td>No</td>
</tr>
<tr>
<td>Create User Service</td>
<td>No</td>
</tr>
<tr>
<td>Delete User Service</td>
<td>No</td>
</tr>
<tr>
<td>Get User List Service</td>
<td>No</td>
</tr>
<tr>
<td>Get Role List Service</td>
<td>No</td>
</tr>
<tr>
<td><strong>Data Services</strong></td>
<td></td>
</tr>
<tr>
<td>Text Search Service</td>
<td>No</td>
</tr>
<tr>
<td>Get Repository List Service</td>
<td>No</td>
</tr>
<tr>
<td>Initiate Synchronization Service</td>
<td>No</td>
</tr>
<tr>
<td>Get Data Source List Service</td>
<td>No</td>
</tr>
<tr>
<td>Upload Metadata Service</td>
<td>No</td>
</tr>
<tr>
<td>Extract Data Service</td>
<td>No</td>
</tr>
<tr>
<td><strong>Content Services</strong></td>
<td></td>
</tr>
</tbody>
</table>
An example of a web service accepting multiple entities is Work Item Reassign web service. A sample request is as below:

```xml
<!-- multiple entities in one transaction is not supported for Reassign. Specify one entity per transaction. -->
<!-- multiple transactions are supported -->
<Transaction>
  <Command type="Reassign">
    <Workitem etype="Entity">
      <ExternalKeys>
        <Key name="WORKITEMREFERENCE" type="string">45010</Key>
        <Key name="USER" type="string">del</Key>
      </ExternalKeys>
    </Workitem>
  </Command>
</Transaction>
<Transaction>
  <Command type="Reassign">
    <Workitem etype="Entity">
      <ExternalKeys>
        <Key name="WORKITEMREFERENCE" type="string">43010</Key>
        <!-- delegation is set so that workitems will be delegated to 2 users -->
        <Key name="USER" type="string">sup</Key>
      </ExternalKeys>
    </Workitem>
  </Command>
</Transaction>
```

A summary of web services accepting multiple entities is shown in Table 18:

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Does it accept multiple entities?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Service</td>
<td>No</td>
</tr>
<tr>
<td>Event Management Services</td>
<td></td>
</tr>
<tr>
<td>Get Event Info Service</td>
<td>No</td>
</tr>
<tr>
<td>Get Event Details Service</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 18  Web Services Accepting Multiple Entities
Chapter 2  Record Services

This chapter describes the usage and customization of Record services.

Topics

- CRUD Services for Record Services on page 48
- Record Query Service, page 61
- Record Query Metadata Service, page 84
- Record Query ValidValues (Init) Service, page 88
- Record Validate Service, page 94
- Record Add or Modify Services, page 96
- Record Delete Service, page 105
- Initiate Workflow Service, page 109
- Transaction Atomicity for Record Web Services, page 112
CRUD Services for Record Services

This section describes the operation and features of WSDL Generator Plug-in. The WSDL Generator Plugin accepts a repository file as an input, and generates an output WSDL and corresponding Java source code. The code is then compiled to a deployable web services entity (.aar) file.

Figure 3  WSDL Generator Plugin Process
The generation is offered through Wizard, which is provided as a 'New Project' wizard in the repository designer. The WSDL generation wizard collects the following inputs from the user:

- Source code (Java) package name
- The user selected Repository
- Output Project name

The WSDL should have the following properties:

- Use a single namespace for request and response
- Use the document literal flavor
- Use a SOAP fault for conditions

The schema should have the following properties:

- Define XSD types for each attribute
- Use a close mapping of XML Schema base types to TIBCO MDM base types
- Account for further type restrictions such as length
- Support multi-value attributes

Repositories with FED attributes

The implementation for performing CRUD operations on repositories with a timestamp type attribute mapped to the EFFECTIVEDATE attribute is as follows:

- **Add operation:** You can add any record including future effective date records to a repository.
- **Update and Delete operations:** You can update and delete all records including future effective date records.
- **Find operation:** You can search only those records which are confirmed and already effective. For the successful Find operation, value of the EFFECTIVEDATE attribute should be less than the current system date.

Generating WSDL

Using the eclipse wizard, you can generate the WSDL.
1. Go to **File > New >** and click **Other**.
2. The **Select a wizard** screen is displayed.

3. Select **MDM WSDL Generator** from the **WSDL Generator** tree node.

4. Click **Next**. The Wizard displays the **WSDL Generator** screen.
5. Click **Browse** and select the path where the input file exists. All the *.rep files are populated.

6. Select the appropriate *.rep file from the **Select File** drop-down list. Based on the selected repository file, the repositories are displayed in the **Select Repository** drop-down list.

7. Select a repository from the **Select Repository** drop-down list and click **Next**. The **New Project** screen is displayed.

8. Enter the appropriate project name in the **Project Name** field. By default, the project name is *repository name* _webservices_.

![New Project Screen](image-url)
9. Click Next. The New Project screen is displayed.

10. Select the relationship depth, by default three levels of relationship is supported.
11. Click **Next**. The **Generation Options** screen is displayed.

12. Enter the appropriate package name in the **Package Name** field. By default, a package name is provided. The package name convention is `com.tibco.mdm.integration.webservice.<repository name>_webservice`. You can modify the package name.

13. Click **Finish**.

**WSDL and XML Schema Generation**

During the WSDL generation process, the following four files are created:

- WSDL file
- Request XSD file
- Response XSD file
- Generic Type file

**WSDL file**

The WSDL file contains the primary contract with the new service definition and the four primary operations (add, delete, update, and find). It uses the interoperable document literal representation.
Request XSD file

The Request contains a type definition of the entire repository and for each attribute of the repository. This type should have the type mappings as described in table XSD Types, page 55, including the TIBCO MDM property length which should map for example to xsd:maxLength for a string. The three write operations (add, modify, delete) will have an entire record as input, only the read operation (find) will have a record id as input - and return the entire record as response.

The types definition are defined in the repository model. If you want to change the type definition it must changed at the repository model level.

Response XSD file

The Response file includes the request type definitions and therefore does not need to create many dynamic types. Only for the Response itself, it will create a type for all four operations. Furthermore, the ResultType and EventType is contained in there which is used by the three write operations responses.

Generic Type file

The generic type file contains the UserInfo type.

Table 19  XSD Types

<table>
<thead>
<tr>
<th>Type Name</th>
<th>Type Mapping</th>
<th>Type Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDM BOOLEAN</td>
<td>Xsd:boolean</td>
<td>-</td>
</tr>
<tr>
<td>MDM Integer</td>
<td>Xsd:integer</td>
<td>-</td>
</tr>
<tr>
<td>MDM Decimal</td>
<td>Xsd:decimal</td>
<td>MDM length, xsd:fractionDigits</td>
</tr>
<tr>
<td>MDM String</td>
<td>Xsd:string</td>
<td>MDM length, xsd:maxInclusive</td>
</tr>
<tr>
<td>MDM File</td>
<td>Xsd:string</td>
<td>Xsd:maxLength = 255</td>
</tr>
<tr>
<td>MDM Amount</td>
<td>Xsd:decimal</td>
<td>MDM length, xsd:fractionDigits</td>
</tr>
<tr>
<td>MDM Custom Decimal</td>
<td>Xsd:decimal</td>
<td>MDM Length, Xsd:fractionDigits</td>
</tr>
<tr>
<td>MDM Date</td>
<td>Xsd:dateTime</td>
<td></td>
</tr>
</tbody>
</table>
CRUD Operations

The CRUD operations refer to Create, Retrieve, Update, and Delete operations. These are the most common operations that are performed on a repository. The following sample depicts the requests for all four operations for the Phone repository:

1. Add/Create
   - Identified by ID and/or IDExt
   - Exception: Record already added

```xml
<ns:AddPersonRequest>
  <!--Optional:-->
  <ns:UserInfo>
    <ns:UserName>admin</ns:UserName>
    <ns:Password>admin</ns:Password>
    <ns:Enterprise>rbs</ns:Enterprise>
  </ns:UserInfo>
  <!--Optional:-->
  <ns:Context>
    <!--Optional:-->
    <ns:Validation>true</ns:Validation>
    <!--Optional:-->
    <ns:Process>true</ns:Process>
    <!--Optional:-->
    <ns:SystemAttributeReturn>false</ns:SystemAttributeReturn>
    <!--Optional:-->
    <ns:ReturnFileAsAttachment>false</ns:ReturnFileAsAttachment>
    <!--Optional:-->
    <ns:EffectiveDateSearchOption>E</ns:EffectiveDateSearchOption>
    <!--Optional:-->
    <ns:DeleteAllRelationships>false</ns:DeleteAllRelationships>
    <!--Optional:-->
    <ns:DeleteAllRelatedRecords>false</ns:DeleteAllRelatedRecords>
  </ns:Context>
  <ns:Person>
    <ns:ID>wt3</ns:ID>
    <!--Optional:-->
    <ns:IDEXT>333</ns:IDEXT>
    <!--Optional:-->
    <ns:CONTAINS>asdf</ns:CONTAINS>
    <!--Optional:-->
    <ns:firstName>user</ns:firstName>
    <!--Optional:-->
    <ns:lastName>wthree</ns:lastName>
    <!--Optional:-->
    <ns:DOB>2000-01-01</ns:DOB>
    <!--Optional:-->
    <ns:phoneNumber>7772121111</ns:phoneNumber>
    <!--Optional:-->
    <ns:PersonToAddressList>
      <!--1 or more repetitions:-->
      <ns:PersonToAddress>
        <ns:IsPrimary>true</ns:IsPrimary>
      </ns:PersonToAddress>
    </ns:PersonToAddressList>
  </ns:Person>
</ns:AddPersonRequest>
```
<ns:Address>
  <ns:ID>wa3</ns:ID>
  <!--Optional:-->
  <ns:IDEXT>333</ns:IDEXT>
  <!--Optional:-->
  <ns:CONTAINS>sdsd</ns:CONTAINS>
  <!--Optional:-->
  <ns:Name>user, wthree</ns:Name>
  <!--Optional:-->
  <ns:streetAddress>222, street address</ns:streetAddress>
  <!--Optional:-->
  <ns:city>this city</ns:city>
  <!--Optional:-->
  <ns:zipCode>11111</ns:zipCode>
</ns:Address>
</ns:PersonToAddress>
</ns:PersonToAddressList>
</ns:Person>
</ns:AddPersonRequest>

2. Modify/Update

- Identified by ID and/or IDExt
- Exception: Record not found

<ns:UpdatePersonRequest>
  <!--Optional:-->
  <ns:UserInfo>
    <ns:UserName>admin</ns:UserName>
    <ns:Password>admin</ns:Password>
    <ns:Enterprise>rbs</ns:Enterprise>
  </ns:UserInfo>
  <!--Optional:-->
  <ns:Context>
    <!--Optional:-->
    <ns:Validation>true</ns:Validation>
    <!--Optional:-->
    <ns:Process>true</ns:Process>
    <!--Optional:-->
    <ns:SystemAttributeReturn>false</ns:SystemAttributeReturn>
    <!--Optional:-->
    <ns:ReturnFileAsAttachment>false</ns:ReturnFileAsAttachment>
    <!--Optional:-->
    <ns:EffectiveDateSearchOption>E</ns:EffectiveDateSearchOption>
    <!--Optional:-->
    <ns:DeleteAllRelationships>false</ns:DeleteAllRelationships>
    <!--Optional:-->
    <ns:DeleteAllRelatedRecords>false</ns:DeleteAllRelatedRecords>
  </ns:Context>
  <ns:Person>
    <ns:ID>wt3</ns:ID>
    <!--Optional:-->
    <ns:IDEXT>333</ns:IDEXT>
    <!--Optional:-->
  </ns:Person>
</ns:UpdatePersonRequest>
<ns:CONTAINS>asdf</ns:CONTAINS>
<!--Optional:-->  
<ns:firstName>user</ns:firstName>
<!--Optional:-->  
<ns:lastName>w3th</ns:lastName>
<!--Optional:-->  
<ns:DOB>1978-01-01</ns:DOB>
<!--Optional:-->  
<ns:phoneNumber></ns:phoneNumber>
<!--Optional:-->  
<ns:PersonToAddressList>
<!--1 or more repetitions:-->  
<ns:PersonToAddress>
<ns:IsPrimary>true</ns:IsPrimary>
<ns:Address>
<ns:ID>aw3</ns:ID>
<!--Optional:-->  
<ns:IDEXT>333</ns:IDEXT>
<!--Optional:-->  
<ns:CONTAINS></ns:CONTAINS>
<!--Optional:-->  
<ns:Name>user, dummy</ns:Name>
<!--Optional:-->  
<ns:streetAddress>111, dummy street</ns:streetAddress>
<!--Optional:-->  
<ns:city>this city</ns:city>
<!--Optional:-->  
<ns:zipCode>11221</ns:zipCode>
</ns:Address>  
</ns:PersonToAddress>
</ns:PersonToAddressList>
</ns:Person>  
</ns:UpdatePersonRequest>  

3. Delete  

- Identified by ID and/or IDExt  
- Exception: Record not found  

<ns:DeletePersonRequest>
<!--Optional:-->  
<ns:UserInfo>
<ns:UserName>?</ns:UserName>
<ns:Password>?</ns:Password>
<ns:Enterprise>?</ns:Enterprise>
</ns:UserInfo>
<!--Optional:-->  
<ns:Context>
<!--Optional:-->  
<ns:Validation>?</ns:Validation>
<!--Optional:-->  
<ns:Process>?</ns:Process>
<!--Optional:-->  
<ns:SystemAttributeReturn>?</ns:SystemAttributeReturn>
<!--Optional:-->  

TIBCO MDM Web Services Guide
4. **Find/Retrieve**

- Unique mode: ID and IDExt.
- Specified -> Single record return

```xml
<ns:FindPersonRequest>
  <!--Optional:-->
  <ns:UserInfo>
    <ns:ReturnFileAsAttachment>?</ns:ReturnFileAsAttachment>
    <!--Optional:-->
    <ns:EffectiveDateSearchOption>?</ns:EffectiveDateSearchOption>
    <!--Optional:-->
    <ns:DeleteAllRelationships>?</ns:DeleteAllRelationships>
    <!--Optional:-->
    <ns:DeleteAllRelatedRecords>?</ns:DeleteAllRelatedRecords>
  </ns:Context>
  <ns:Person>
    <ns:ID>?</ns:ID>
    <!--Optional:-->
    <ns:IDEXT>?</ns:IDEXT>
    <!--Optional:-->
    <ns:CONTAINS>?</ns:CONTAINS>
    <!--Optional:-->
    <ns:firstName>?</ns:firstName>
    <!--Optional:-->
    <ns:lastName>?</ns:lastName>
    <!--Optional:-->
    <ns:DOB>?</ns:DOB>
    <!--Optional:-->
    <ns:phoneNumber>?</ns:phoneNumber>
    <!--Optional:-->
    <ns:PersonToAddressList>
      <!--1 or more repetitions:-->
      <ns:PersonToAddress>
        <ns:IsPrimary>?</ns:IsPrimary>
        <ns:Address>
          <ns:ID>?</ns:ID>
          <!--Optional:-->
          <ns:IDEXT>?</ns:IDEXT>
          <!--Optional:-->
          <ns:CONTAINS>?</ns:CONTAINS>
          <!--Optional:-->
          <ns:Name>?</ns:Name>
          <!--Optional:-->
          <ns:streetAddress>?</ns:streetAddress>
          <!--Optional:-->
          <ns:city>?</ns:city>
          <!--Optional:-->
        </ns:Address>
      </ns:PersonToAddress>
    </ns:PersonToAddressList>
  </ns:Person>
</ns:DeletePersonRequest>
```
<ns:UserName>admin</ns:UserName>
<ns:Password>admin</ns:Password>
<ns:Enterprise>rbs</ns:Enterprise>
</ns:UserInfo>
<!--Optional:-->
<ns:Context>
  <!--Optional:-->
  <ns:Validation></ns:Validation>
  <!--Optional:-->
  <ns:Process></ns:Process>
  <!--Optional:-->
  <ns:SystemAttributeReturn></ns:SystemAttributeReturn>
  <!--Optional:-->
  <ns:ReturnFileAsAttachment>false</ns:ReturnFileAsAttachment>
  <!--Optional:-->
  <ns:EffectiveDateSearchOption></ns:EffectiveDateSearchOption>
  <!--Optional:-->
  <ns:DeleteAllRelationships></ns:DeleteAllRelationships>
  <!--Optional:-->
  <ns:DeleteAllRelatedRecords></ns:DeleteAllRelatedRecords>
</ns:Context>
<!--1 or more repetitions:-->
<ns:FindPersonID>
  <ns:ID>wt3</ns:ID>
  <!--Optional:-->
  <ns:IDEXT>333</ns:IDEXT>
</ns:FindPersonID>
</ns:FindPersonRequest>
Record Query Service

Use the Record Query web service to query one or more records. You can even query the deleted records.

The response returns attribute values and relationships to other records. The query is based on one or more `<ExternalKeys>` specified in the request.

Record Query Service - Inputs

Command Type

Command type needs to be set to Query.

```xml
<Command type="Query">
<MasterCatalogRecord etype="Entity">
</MasterCatalogRecord>
</Command>
```

Query Directives

Pagination

To achieve pagination of a large result, specify the following tags:

- `<MaxCount>`: Limits the number of records returned, if more than one record matches the query.
- `<StartCount>`: In combination with MaxCount, the caller can control which 'page' of the result set needs to be returned.
- `<OrderByColumnList>`: Defines order by Column list with sort order for the results. This tag is Optional.

You can retrieve data in sort order based on one or more columns in an ascending or descending order.

Example 1 Query Directives

```xml
<Command type="Query">
  <StartCount>5</StartCount>
  <MaxCount>50</MaxCount>
  <!-- Sample OrderByColumnList syntax is -->
  <!-- <OrderByColumnList>AttributeName1,DESC:AttributeName2,ASC:AttributeName3</OrderByColumnList>  -->
  <!-- Order can be ASC or DESC, and is optional. Default being ASC -->
  <OrderByColumnList>PRODUCTID, DESC:CustName, ASC:AmtPaid</OrderByColumnList>
</Command>
```

<!-- Above sort order would first sort on records by 'PRODUCTID' in 'DESCENDING' order, then further sort the results obtained by previous sort by 'CustName' in 'ASCENDING' order and lastly further sort the results by sorting on 'AmtPaid' in default
'ASCENDING' order. Results returned would be further filtered based on StartCount and MaxCount specified -->

```
<Mast erCatalogRecord etype="Entity">
  <ExternalKeys>
    <!-- First Key is Master Catalog name -->
    <Key name="MASTERCATALOGNAME">CUSTOMER</Key>
    <Key name="PRODUCTID">cust_2</Key>
  </ExternalKeys>
</MasterCatalogRecord>
```

- `<FUTURE_DATED_RECORD>`: Queries records by Future Effective Date key options. Specify any one the following key options:
  
  — I (Include FED): Includes Future Effective Date records along with other records.
  
  — O (Only FED): Includes only Future Effective Date records.
  
  — N (No FED): Does not include Future Effective Date records.
  
  Based on these key options, the records are returned. If you do specify any key option, by default the N (No FED) option is retrieved.

**Example 2 Future Dated Record Query I**

```
<Command type="Query">
  <MasterCatalogRecord etype="Entity">
    <ExternalKeys>
      <!-- First Key is Master Catalog name -->
      <Key name="MASTERCATALOGNAME">MASTERCATALOG_WITH_EFFECTIVEDATE_RA</Key>
    </ExternalKeys>
    <Key name="PRODUCTID">1a</Key>
    <!--Key option -- I (Include FED), O (Only FED), N (No FED). If other than I, O, N is passed then it takes the default option i.e. N (No FED/Exclude FED)-->  
    〈Key name="FUTURE_DATED_RECORD">I</Key>
  </ExternalKeys>
</MasterCatalogRecord>
```

In the Record Query request, do not use `FUTURE_DATED_RECORD` and `EFFECTIVEDATE` tags together.

When querying a record, specify the `EFFECTIVEDATE` variable in context to return the related record according to the effective date of a relationship.

**Example 3 Effective Date Related Record Query II**

```
<Context>
  <Variable name="EFFECTIVEDATE">2010-01-01 00:00:00.000</Variable>
</Context>
```
<Command type="Query">
  <MasterCatalogRecord etype="Entity">
    <ExternalKeys>
      <Key name="MASTERCATALOGNAME">SELF9</Key>
      <Key name="PRODUCTID">fed1</Key>
      <Key name="EFFECTIVEDATE">2013-01-01 00:00:00.000</Key>
      <!--Key name="FUTURE_DATED_RECORD">I</Key-->
    </ExternalKeys>
    <Return>
      <RelationshipData>
        <Relationship>
          <RelationType>CUSTOMERADDRESS</RelationType>
        </Relationship>
        <RelationshipData>
        </Return>
      </MasterCatalogRecord>
  </Command>
</Transaction>

Defining query criterion using External Keys

The <Entity> tag specifies how to look up Records in the repository. Currently, the only query method supported is by ExternalKeys. For different Entity types, the ExternalKeys have different semantics. In this scenario, the Entity type is MasterCatalogRecord which has the following semantics:

Table 20  Record Query - External Keys

<table>
<thead>
<tr>
<th>Key Name</th>
<th>Meaning</th>
<th>Optional or Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTERCATALOGNAME</td>
<td>The name of the repository containing records. This is the first key specified in the list. All subsequent keys make up the search criteria for the records in the catalog.</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>
The criterion the record needs to meet to be selected — the name of the attribute and its value needs to be specified.

The keys are specified by using Key tag. It has the following attributes:

- name — Name of the attribute.
- operator — Operator to use for the search.

For example:

```xml
<Key
    name="FIRSTNAME"
    operator="eq">John</Key>
```

**Note:** All external keys provided in the request have to match for a record to be selected.

At this time, ‘AND’ is the only operator allowed for building the search criterion. This is implicit, and no user inputs are required. In other words, the query is:

```
Attribute1 = 'Value1' AND Attribute2 = 'Value2' ...
```
Example 4  Record Query - External Keys

<ExternalKeys>
<!-- First Key is repository name -->
<Key name="MASTERCATALOGNAME">CUSTOMER</Key>
<!-- Subsequent are Record Keys -->
<Key name="FIRSTNAME" operator="eq">John</Key>
</ExternalKeys>

This example returns a list of records from CUSTOMER repository, which have FIRSTNAME specified as John.

You can specify the multi-value attributes in the search criteria similar to the regular attribute. For example:

<ExternalKeys>
<!-- First Key is repository name -->
<Key name="MASTERCATALOGNAME">PRODUCT</Key>
<!-- Subsequent are Record Keys -->
<Key name="TYPE" operator="eq">shirt</Key>
<Key name="SIZE" operator="eq">XL</Key>
</ExternalKeys>

This example returns those products whose TYPE is equal to shirt and one of the value for SIZE is XL, where SIZE is a multi-value attribute.

Predefined Keys

You can specify the following predefined keys along with the repository attributes:

- RECORD_MODDATE: Identifies the modification date of the record. It can take values as:
  - Date: Specify the date in any date format.
  - Operator: For information on the supported list of operators for the Date data type, refer to Record Query - Supported Operators on page 69.

Example 5  RECORD_MODDATE

The following example searches a Customer record with less than the modified date:

<ExternalKeys>
<Key name="MASTERCATALOGNAME" type="string">CUSTOMER</Key>
<!-- <Key name="PRODUCTID" type="string">c1</Key> -->
<Key name="PRODUCTIDEXT" type="string">c1</Key>
<Key name="RECORD_MODDATE" operator="lt">05/05/2011</Key>
</ExternalKeys>
• RECORD_STATE: Identifies the state of a record that is being searched. The RECORD_STATE key supports a list of multiple values separated by a comma. The following values are supported:
  — CONFIRMED: Includes the confirmed records. This is the default value.
  — UNCONFIRMED: Includes the unconfirmed records.
  — REJECTED: Includes the rejected records.
  — DRAFT: Includes the draft records.

The DRAFT record versions are owned by an event. The version is created during the event. To retrieve the DRAFT version records, you must specify the work item context.
  — LATEST: Includes the confirmed or unconfirmed records.
  — ALL: Includes all records with any state.

Specify the LATEST and ALL values to include multiple states of a record.

Example 6  RECORD_STATE with the CONFIRMED value

The following example searches a confirmed record with CITY as San Jose:

```xml
<ExternalKeys>
  <Key name="MASTERCATALOGNAME">ADDRESS</Key>
  <Key name="CITY" operator="eq">San Jose</Key>
  <Key name="RECORD_STATE">CONFIRMED</Key>
</ExternalKeys>
```

Example 7  RECORD_STATE with multiple values

The following example searches a confirmed and unconfirmed records with CITY as San Jose:

```xml
<ExternalKeys>
  <Key name="MASTERCATALOGNAME">ADDRESS</Key>
  <Key name="CITY" operator="eq">San Jose</Key>
  <Key name="RECORD_STATE">CONFIRMED,UNCONFIRMED</Key>
</ExternalKeys>
```
- **RECORD_VERSION**: Identifies the version of a record that is being searched. You can specify the following values:
  - `<integer value>`: A single record query. Indicates the record version to be retrieved.

**Example 8  RECORD_VERSION with <integer> value**

The following example retrieves the data for 12345, SAVING record and for version 5.

```xml
<ExternalKeys>
  <Key name="MASTERCATALOGNAME">ACCOUNT</Key>
  <Key name="PRODUCTID">12345</Key>
  <Key name="PRODUCTIDEXT">SAVING</Key>
  <Key name="RECORD_VERSION">5</Key>
</ExternalKeys>
```

- **ALL**: Retrieves all versions of a record based on the specified criteria.

  In the prior releases, if you had specified `RECORD_VERSION=ALL`, all versions were retrieved in the response except for the DRAFT records. Additionally, the deleted versions were also retrieved.

  From 8.3 release onwards, if you specify `RECORD_VERSION=ALL`, all CONFIRMED and ACTIVE versions are retrieved in the response. However, if the default value of the `RECORD_STATE` and `ACTIVE` keys is used for filtering, the retrieved versions may not be all. Therefore, to retrieve all versions of a record, specify the following multiple combination of keys:

**Example 9  RECORD_VERSION with ALL value**

```xml
<ExternalKeys>
  <Key name="RECORD_VERSION" operator="eq">ALL</Key>
  <Key name="RECORD_STATE">ALL</Key>
  <Key name="ACTIVE" type="string">ANY</Key>
</ExternalKeys>
```

Only the eq operator is supported for the `RECORD_VERSION` key.
- **ACTIVE**: Queries whether or not the deleted records included in the response. The valid values for this key are:
  - **Y**: Only active records are retrieved. Query does not return any deleted record. This is the default.
  - **N**: Only data for deleted records is retrieved. Query does not return any active record.
  - **ANY**: Indicates that both active and deleted records are retrieved, that is both `ACTIVE = Y` and `ACTIVE = N` values are considered.

If you specify the value other than the valid values, `Y` is assumed.

**Example 10  ACTIVE**

This example searches a deleted record with the specified `PRODUCTID` and `EXT`:

```xml
<Transaction>
  <Command type="Query">
    <StartCount>1</StartCount>
    <MaxCount>10</MaxCount>
    <MasterCatalogRecord etype="Entity">
      <ExternalKeys>
        <Key name="MASTERCATALOGNAME" type="string">DELETETEST</Key>
        <Key name="PRODUCTID" operator="eq">TC1</Key>
        <Key name="PRODUCTIDEXT" operator="eq">TC1</Key>
        <Key name="ACTIVE" type="string">N</Key>
      </ExternalKeys>
    </MasterCatalogRecord>
  </Command>
</Transaction>
```

**Example 11  RECORD_STATE and ACTIVE**

This example searches deleted records with the record state as `CONFIRMED`:

```xml
<Transaction>
  <Command type="Query">
    <StartCount>1</StartCount>
    <MaxCount>10</MaxCount>
    <MasterCatalogRecord etype="Entity">
      <ExternalKeys>
        <Key name="MASTERCATALOGNAME" type="string">DELETETEST</Key>
        <Key name="NAME" operator="eq">TC1</Key>
        <Key name="ACTIVE" operator="eq">N</Key>
        <Key name="RECORD_STATE">CONFIRMED</Key>
      </ExternalKeys>
    </MasterCatalogRecord>
  </Command>
</Transaction>
```
### Operators

This section provides a list of supported and deprecated operators.

#### Supported Operators

The following is a list of supported operators:

**Table 21  Record Query - Supported Operators**

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
<th>Data Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>eq</td>
<td>Equals</td>
<td>ALL</td>
</tr>
<tr>
<td>ne</td>
<td>Not equals</td>
<td>ALL</td>
</tr>
<tr>
<td>id</td>
<td>Is Defined</td>
<td>ALL</td>
</tr>
<tr>
<td>nd</td>
<td>Not Defined</td>
<td>ALL</td>
</tr>
<tr>
<td>in</td>
<td>In</td>
<td>ALL except Boolean</td>
</tr>
<tr>
<td>lk</td>
<td>Like</td>
<td>String, File</td>
</tr>
<tr>
<td>nl</td>
<td>Not Like</td>
<td>String, File</td>
</tr>
<tr>
<td>sw</td>
<td>Starts With</td>
<td>String, File</td>
</tr>
<tr>
<td>ew</td>
<td>Ends With</td>
<td>String, File</td>
</tr>
<tr>
<td>sl</td>
<td>Sounds like</td>
<td>String, File</td>
</tr>
<tr>
<td>gt</td>
<td>Greater than</td>
<td>Number, Date, and Timestamp</td>
</tr>
<tr>
<td>ge</td>
<td>Greater than equals</td>
<td>Number, Date, and Timestamp</td>
</tr>
<tr>
<td>lt</td>
<td>Less than</td>
<td>Number, Date, and Timestamp</td>
</tr>
<tr>
<td>le</td>
<td>Less than equals</td>
<td>Number, Date, and Timestamp</td>
</tr>
<tr>
<td>bw</td>
<td>Between</td>
<td>Number, Date, and Timestamp</td>
</tr>
</tbody>
</table>

*Note:* Use comma as value separator and double quotes as text qualifier in case the value itself contains comma.
For the string type attributes, additionally, you can check for case sensitivity using `caseSensitive`. It specifies whether a case sensitive query is to be done. The default is `false`.

**Deprecated Operators**

The following attributes are deprecated. Instead, use the operator attribute to specify the required criteria:

**Table 22  Record Query - Deprecated and Equivalent Operators**

<table>
<thead>
<tr>
<th>Deprecated Operator</th>
<th>Description</th>
<th>Equivalent Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>lowerLimit</td>
<td>Specifies the lower limit of the numeric range. When only the lower limit is specified, all the records with a value greater than and equal to the lower limit are returned.</td>
<td>gt</td>
</tr>
<tr>
<td>upperLimit</td>
<td>Specifies upper limit of the numeric range. When only the upper limit is specified, all the records with a value less than or equal to the upper limit are returned.</td>
<td>lt</td>
</tr>
<tr>
<td>after</td>
<td>Specifies the lower limit of the date range. When only <code>&lt;after&gt;</code> is specified, all the records with a date greater than and equal to the <code>&lt;after&gt;</code> date are returned. <strong>Note:</strong> The <code>DD-MM-YY</code> date format is not supported.</td>
<td>ge</td>
</tr>
<tr>
<td>before</td>
<td>Specifies the upper limit of the date range. When only <code>&lt;before&gt;</code> is specified, all the records with a date less than and equal to the <code>&lt;before&gt;</code> date are returned. <strong>Note:</strong> The <code>DD-MM-YY</code> date format is not supported.</td>
<td>le</td>
</tr>
</tbody>
</table>
Relationship Search Criteria

You can search the record based on:

- Relationship attributes of any relationship defined for current repository
- Existence of related records
- Count of related records (for example, a customer which has two associated banks)
- Attributes from target repository.

The RelationshipData tag defines how to specify the relationship criteria for the repository. For example:

```xml
<Transaction>
  <Command type="Query">
    <StartCount>1</StartCount>
    <MaxCount>1000</MaxCount>
    <MasterCatalogRecord etype="Entity">
      <!-- Base master catalog attributes -->
      <ExternalKeys>
        <Key name="MASTERCATALOGNAME" type="string">Customer</Key>
        <Key name="PRODUCTID">a2</Key>
        <Key name="PRODUCTIDEXT">a2</Key>
        <Key name="FIRSTNAME">John</Key>
      </ExternalKeys>
      <!-- Relationship search criteria -->
      <RelationshipData>
        <!-- flags exists , count -->
        <Relationship checkExistence ="true" count="2">
          <RelationType>CustomerAccount</RelationType>
      </RelationshipData>
  </MasterCatalogRecord>
</Command>
</Transaction>
```

Table 22  Record Query - Deprecated and Equivalent Operators

<table>
<thead>
<tr>
<th>Deprecated Operator</th>
<th>Description</th>
<th>Equivalent Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>exact</td>
<td>Specifies whether an exact match of the string needs to be done. When &lt;exact&gt; is set to true, an exact match of the string is returned. By adding the ‘*’ or ‘%’ character to the external keys, a SQL ‘LIKE’ equivalent comparison can also be performed.</td>
<td>Instead of exact=&quot;true&quot;, use operator =&quot;eq&quot;. Instead of exact=&quot;false&quot;, use operator=&quot;lk&quot;.</td>
</tr>
</tbody>
</table>
<!-- Relationship attributes -->
<RelationshipAttributes>
    <Attribute name="type">SAVING</Attribute>
</RelationshipAttributes>

<!-- target master catalog attributes.-->
<Record>
    <ExternalKeys>
        <Key name="BALANCE" >39000</Key>
    </ExternalKeys>

    <!-- second level relationship search -->
    <RelationshipData>
        <Relationship checkExistence ="false">
            <RelationType>AccountAddress</RelationType>
            <RelationshipAttributes>
                <Attribute name="preferred">Y</Attribute>
            </RelationshipAttributes>
            <Record>
                <ExternalKeys>
                    <Key name="CITY">San Jose</Key>
                </ExternalKeys>
            </Record>
        </Relationship>
    </RelationshipData>
</Record>
</MasterCatalogRecord>
</Command>
</Transaction>

Table 23   Record Query - Keys

<table>
<thead>
<tr>
<th>Key Name</th>
<th>Meaning</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>RelationType</td>
<td>Relationship name</td>
<td>Yes</td>
</tr>
<tr>
<td>checkExistence</td>
<td>This is optional flag which allows to search based on existence or non-existence of a relationship. Valid values are true, false (means does not exists)</td>
<td>No. Default is Yes.</td>
</tr>
<tr>
<td>count</td>
<td>Count of related records</td>
<td>No</td>
</tr>
</tbody>
</table>
Optimization Flags for Record Query

- `com.tibco.cim.optimization.recordsearch.relationship.depth` — The records can be searched based on multiple or multi-level relationships. This flag specifies maximum depth or width of the relationship for which record search can be executed.

- `com.tibco.cim.optimization.recordsearch.maxjoins` — Maximum number of SQL joins allowed in record search. Each inclusion of multi-value attribute in search is counted as one join. Each inclusion of relationship attribute in search is counted as one join.

### Table 23  Record Query - Keys

<table>
<thead>
<tr>
<th>Key Name</th>
<th>Meaning</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>RelationshipAttributes</td>
<td>The criterion for relationship attributes can be specified using Attribute key. For example: <code>&lt;Attribute name=&quot;ACCOUNT_TYPE&quot;&gt;SAVING&lt;/Attribute&gt;</code></td>
<td>No</td>
</tr>
<tr>
<td>Record</td>
<td>The criterion for target repository attributes can be specified using ExternalKeys tag (the name of the attribute and its value needs to be specified). For example: <code>&lt;Record&gt;&lt;ExternalKeys&gt;&lt;Key name=&quot;CITY&quot;&gt;San Jose&lt;/Key&gt;&lt;/ExternalKeys&gt;&lt;/Record&gt;</code></td>
<td>No</td>
</tr>
</tbody>
</table>
Querying Single Record

The query can be made by specifying `PRODUCTID` and `PRODUCTIDEXT`. An unique record is returned.

```xml
<ExternalKeys>
  <Key name="MASTERCATALOGNAME">CUSTOMER</Key>
  <Key name="PRODUCTID">6001</Key>
  <Key name="PRODUCTIDEXT">P_69_6001</Key>
</ExternalKeys>
```

Handling File Type Attributes

TIBCO MDM returns file type attributes either as attachments or as hyperlinks. This is done by providing a `FILE_TYPE_ATTRIBUTES` context variable. It has the following valid values:

- ATTACHMENT — This indicates that the client would like the server to return the file type attributes as SOAP attachments to the web services response.
- HREF — This indicates that the client would like the server to return a hyperlink into the application, from where client can download the files represented by the file type attributes. This is the default.

For example:

```xml
<Context>
  <Variable name="FILE_TYPE_ATTRIBUTES" type="string">HREF</Variable>
</Context>
```

Return

Selecting the list of attributes — The list of attribute returned in the response can be filtered based on output map define in the repository. When the output map is specified in the request, only those attributes from the repository that map to the output map are returned in the response.

```xml
<Return>
  <!-- CatalogOutputMap specifies which attributes to output -->
  <Entity type="OutputMap">
    <ExternalKeys>
      <Key name="name">ProductCatalogCase</Key>
    </ExternalKeys>
    <Context>
      <!-- CatalogOutputMap specifies which attributes to output -->
      <Variable name="FILE_TYPE_ATTRIBUTES" type="string">HREF</Variable>
    </Context>
  </Entity>
</Return>
```

This `<Return>` section tells the system to output those attributes specified in the Catalog Output Map called `ProductCatalogCase`.
- Selecting the list of Relationships — As part of the <Return> section, you can also specify a list of related records you are interested in. This is done by specifying the list of relationship types.

**Note:** The full hierarchy for the selected relationship types is returned. For example:

```
<Return>
  <RelationshipData>
    <Relationship>
      <RelationType>Contains</RelationType>
    </Relationship>
    <Relationship>
      <RelationType>ContainedBy</RelationType>
    </Relationship>
  </RelationshipData>
</Return>
```

**Querying Records for a Specific Work Item Context**

To query records for a specific work item context, specify an additional WORKITEMREFERENCE input parameter as shown in the following example:

```
<DataService version="2.0">
  <Identity>
    <DirectoryPath>
      <Directory type="Enterprise">${#Project#Enterprise}</Directory>
      <Directory type="User">${#Project#User}</Directory>
    </DirectoryPath>
    <Authentication>${#Project#password}</Authentication>
  </Identity>
  <Transaction>
    <Context>
      <Variable name="WORKITEMREFERENCE" type="string">37130</Variable>
    </Context>
    <Command type="Query">
      <StartCount>1</StartCount>
      <MaxCount>10</MaxCount>
      <MasterCatalogRecord etype="Entity">
        <ExternalKeys>
          <Key name="MASTERCATALOGNAME" type="string">${#Project#MASTERCATALOGNAME}</Key>
          <Key name="PRODUCTID" operator='eq'>REC1</Key>
          <Key name="PRODUCTIDEXT" operator='eq'>REC1</Key>
        </ExternalKeys>
      </MasterCatalogRecord>
    </Command>
  </Transaction>
</DataService>
```

The “Query records for workitem” does not return related records of the given record. For example, if there is a hierarchy a-contains-b-contains-c and you have queried for record “a”, only the matching record for the given filter criteria is returned in the response XML.
Querying Records Based on Classifications

You can query records based on a classification scheme or code. To do this, define the classification scheme name and code as a context variable in the request.

- ClassificationScheme — Specify the classification scheme name.
- ClassificationCode — Specify the classification code to retrieve records classified in that code.
- To get unclassified records, specify classification scheme name.

Only the eq operator is supported for classification code and scheme. If you do not specify the classification scheme, the classification code is ignored.

The following is a request example:

```xml
<Transaction>
  <Context>
    <Variable name="CLASSIFICATIONSCHEME" type="string">LOCATION</Variable>
    <Variable name="CLASSIFICATIONCODE" type="string">INDIA</Variable>
  </Context>
  <Command type="Query">
    <StartCount>1</StartCount>
    <MaxCount>10</MaxCount>
    <MasterCatalogRecord etype="Entity">
      <ExternalKeys>
        <Key name="MASTERCATALOGNAME">ADDRESS</Key>
        <Key name="HOUSE_NO" operator="eq">105</Key>
      </ExternalKeys>
    </MasterCatalogRecord>
  </Command>
</Transaction>
```

Returning Record Classification Details

To return record classification details, you need to specify the <ClassificationSchemes> context in the <Return> tag. You can search for records including the record classification details.

- ClassificationScheme — Specify one or more classification scheme names in the <ClassificationSchemes> context, which includes the classification and record information.

```xml
<Return>
  <ClassificationData>
    <ClassificationSchemes>
      <ClassificationScheme name="food"/>
    </ClassificationSchemes>
  </ClassificationData>
</Return>
```
The sample XML files are available at the following locations:

- `$MQ_HOME\schema\DataService\2.0\samples\repository\record-query\MasterCatalogRecord-QueryData-ReturnClassificationDetails-Request.xml`
- `$MQ_HOME\schema\DataService\2.0\samples\repository\record-query\MasterCatalogRecord-QueryData-ReturnClassificationDetails-Response.xml`

**Querying Records to Filter Null Value Attributes**

By default, all attributes are displayed as a part of web service response. However, if you want to filter the null value attributes from the response, specify `true` for the `SUPPRESS_NULL_EMPTY_ATTRIBUTES` context variable in the record query request. The variable filters attributes with the null or empty values from the web service query responses. You can use the variable for the following record operations:

- Record search
- Related record search or relationship search with relationship attributes
- Records with category specific attributes
- Record search for deleted records
- Record query based on classification data
- Record query based on valid values
- Record query using output map

**Example 12  Record Query Request File**

```xml
<ns:Context>
  <ns:Variable name="SUPPRESS_NULL_EMPTY_ATTRIBUTES" type="string">true</ns:Variable>
</ns:Context>
```
Querying Records Based on Context Variables Values

You can query for records by passing custom context variable as name value pair to Rulebase. The context variable can be used in condition to filter records, the context variable should be declared as usage="local" in TIBCO MDM Studio.

Example 13  Filtering Records Based on Effective Date using Context Variables

Declare RUL_EFF_DATE1 and RUL_SHORTDESC variables in TIBCO MDM Studio Rulebase Designer; specify usage="local".

Refer the Rulebase sample:

```xml
<rulebase metaversion="1.0" xmlns="http://www.tibco.com/cim/rulebase/beans">
  <name>rule</name>
  <declare usage="local">
    <var>RUL_EFF_DATE1</var>
    <datatype type="date"/>
  </declare>
  <declare usage="local">
    <var>RUL_SHORTDESC</var>
    <datatype type="string"/>
  </declare>
  <constraint>
    <name>con</name>
    <description>con</description>

    <condition>
      <and>
        <eq>
          <var>EFFECTIVEDATE</var>
          <var>RUL_EFF_DATE1</var>
        </eq>
        <eq>
          <var>SHORTDESC</var>
          <var>RUL_SHORTDESC</var>
        </eq>
      </and>
    </condition>
  </constraint>
  <action>
    <access mode="hide_record"/>
  </action>
</rulebase>
```
Pass the context variable in webservice request as below:

```xml
<DataService version="2.0">
  <Identity>
    <DirectoryPath>
      <Directory type="Enterprise">Tech</Directory>
      <Directory type="User">jsmith</Directory>
    </DirectoryPath>
    <Authentication>a</Authentication>
  </Identity>
  <Transaction>
    <Context>
      <Variable name="RUL_EFF_DATE" type="date">07/11/2011</Variable>
      <Variable name="RUL_SHORTDESC" type="string">RECORD7</Variable>
    </Context>
    <Command type="Query">
      <StartCount>1</StartCount>
      <MaxCount>50</MaxCount>
      <MasterCatalogRecord etype="Entity">
        <ExternalKeys>
          <Key name="MASTERCATALOGNAME" type="string">CUSTOMER</Key>
        </ExternalKeys>
      </MasterCatalogRecord>
    </Command>
  </Transaction>
</DataService>
```

This hides all records with EFFECTIVEDATE="2011-07-11 00:00:00.000" and SHORTDESC =Record7.

**Querying Related Records**

You can query for related records for a given Parent record and relationship name. You can also specify a paging element using the `<StartCount>` and `<MaxCount>` elements.

To query for related records:

1. Specify the user name, password, and enterprise for a user in the request criteria.

2. Specify Command="Query" and command qualifier="RelatedRecords".

---

TIBCO MDM  Web Services Guide
3. Specify the following mandatory input parameters in the `<ParentRecordData>` element:
   - Repository Name
   - ProductID and ProductIDExt
   - UOM
   - Record version
   - Relationship Name

   (OR)

   - Repository Name
   - ProductKeyID
   - Record version
   - Relationship Name

   For example,

   ```xml
   <ParentRecordData>
   <Record>
     <ExternalKeys>
       <Key name="MASTERCATALOGNAME" type="string">CUSTOMER</Key>
       <Key name="PRODUCTID">a1-relA</Key>
       <Key name="PRODUCTIDEXT">a1-relA</Key>
       <!--Key name="PRODUCTKEYID">10001</Key-->
       <Key name="RECORD_VERSION">3</Key>
     </ExternalKeys>
   </Record>
   </ParentRecordData>

4. Specify the relationship details in the `<RelationshipData>` element:

   - `<RelationType>`: Specify the Relationship name to return related records for the specific relationship or specify **ALL** to return all the relationships related to the record.

   - `<RelationDepth>`: Specify the relationship depth in numeric to control the depth of the bundle heirarchy.

   For example,

   ```xml
   <Return>
   <RelationshipData>
     <Relationship>
       <RelationType>CUSTOMERADDRESS</RelationType>
       <RelationDepth>1</RelationDepth>
     </Relationship>
     <Relationship>
       <RelationType>ALL</RelationType>
       <RelationDepth>3</RelationDepth>
     </Relationship>
   </RelationshipData>
   ```
5. Invoke the webservice by specifying the endpoint URL as:
   http://<servername>:<port>/eml/services/router/MasterCatalogAction

   The Response message contains a list of related records.

**Querying Records Based On Perspectives**

You can query for records bundled into a particular perspective. The perspective name must be passed as one of the context attributes and depending on the type of relationship and the depth of relationship. The following table provides the details of relationship type:

**Table 24  Relationship Type for Perspective and Description**

<table>
<thead>
<tr>
<th>Relationship Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>All the records bundled into the perspective until the specified depth are returned.</td>
</tr>
<tr>
<td>RelationshipName which is in the perspective</td>
<td>The related records pertaining to that relationship are returned.</td>
</tr>
<tr>
<td>RelationshipName which is not in the perspective</td>
<td>Only the root record is returned.</td>
</tr>
</tbody>
</table>

A request example is shown below:

```xml
<DataService version="2"
   <Identity>
      <DirectoryPath>
         <Directory type="Enterprise">Tech</Directory>
         <Directory type="User">jsmith</Directory>
      </ns0:DirectoryPath>
      <Authentication type="Password">jsmith</Authentication>
   </Identity>
   <Context>
      <Variable name="PERSPECTIVENAME" type="string">personalInfoPersp</Variable>
   </Context>
   <Transaction>
      <Command type="Query">
```

}}

```xml
</Command>
</Transaction>
</Context>
</Identity>
</DataService>
```
<MaxCount>10</MaxCount>
<MasterCatalogRecord etype="Entity">
  <ExternalKeys>
    <Key name="MASTERCATALOGNAME">PERSON</Key>
    <Key name="PRODUCTID">p1</Key>
  </ExternalKeys>
  <Return>
    <RelationshipData>
      <Relationship>
        <RelationType>ALL</RelationType>
        <RelationDepth>3</RelationDepth>
      </Relationship>
    </RelationshipData>
  </Return>
</MasterCatalogRecord>
</Command>
</Transaction>
</DataService>

Record Query Service - Outputs

Record Data
Each record that matches the filter criterion given in the request is returned as a separate ‘MasterCatalogRecord’. The relationships between the records are captured by the ‘RelationshipData’ section within each ‘MasterCatalogRecord’.

The ‘EntityData’ section contains the attribute values for each record. In case a output map was specified, the attribute list returned matches the attributes in the output map.

Transaction Result
For more information, refer to, TransactionResult, on page 7.

Record Query Service - Access Configuration
The following access configurations can change the behavior of the Record Query service.
Rulebase Configuration

- Value based security – If a certain record is marked out of bounds of a profile, these records are not returned as part of the results.

- Access to attributes – Certain attributes and attribute groups can be marked hidden using rulebase configurations. If no outputmap was specified as part of the query request, a subset of attributes to which the user has access are returned. However, if an output map was specified in the request, a successful response is sent only if the user has access to all repository attributes that were mapped to the output map. If any output map attributes are marked hidden, an authorization error is returned.
Record Query Metadata Service

Use this web service to obtain repository record metadata. The metadata contains information, such as, the name and description of attribute groups. It also contains information about all attributes, such as, name, description, data type, length, and so on. The Record Query Metadata web service also shows dependencies between attributes.

The results of this web service are independent of the data contained in a record.

Record Query MetaData - Inputs

Command type

The Command type needs to be set to Query; the entity command qualifier is set to MetaData.

<Command type="Query">
<MasterCatalogRecord commandqualifier="MetaData" etype="Entity">

Query Directives

NONE

Support for Bundles

This service does not support record bundles.

External Keys

The external keys have the following semantics:

Table 25  Record Query MetaData - External Keys

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTERCATALOGNAME</td>
<td>The name of the repository for which meta-data is queried.</td>
<td>Y</td>
</tr>
</tbody>
</table>

Entity Data

No entity data is supported in the query.
Record Action Intended

You can specify the action that you want to perform on the data that is being queried. The valid record actions are:

- ADD
- EDIT
- COPY
- VIEW
- MASS_UPDATE
- RECORD_SEARCH

You can specify the record action in the Context section of the request. For example,

```xml
<Context>
  <Variable name="RECORD_ACTION" type="string">ADD</Variable>
</Context>
```

For more details on RECORD_ACTION, refer to Appendix A Context Variables in TIBCO MDM Studio Rulebase Designer User’s Guide.

Support for Classification Context

You can retrieve a list of category specific attributes along with other attributes while querying metadata by specifying a classification context. Specify a list classification schemes and code hierarchy in the `<ClassificationSchemes>` context.

By default, category specific attributes are not included. You must specify a valid classification context to return category specific attributes in response.

```xml
<ClassificationSchemes>
  <ClassificationScheme name="FBT-SCHEME1">
    <ClassificationCode code="50201600">
      <ClassificationCode code="50201700">
        <ClassificationCode code="10000111"/>
        <ClassificationCode code="10000115"/>
      </ClassificationCode>
    </ClassificationCode>
  </ClassificationScheme>
</ClassificationSchemes>
```
Classification Scheme — Each classification scheme always has one root code. If you want to locate all category specific attributes, which apply to a classification scheme, specify only one level root code.

ALL — Specify to return all category specific attributes. For example,

```xml
<ClassificationSchemes>
  <All/>
</ClassificationSchemes>
```

If you do not specify `<ClassificationSchemes>` context, the category specific attributes are not returned.

The sample XML files are available at

$MQ_HOME\schema\DataService\2.0\samples\repository\metadata:
- RecordQueryMetaDataWithClassificationContextRequest.xml
- RecordQueryMetaDataWithClassificationContextResponse.xml
- RecordQueryMetaDataWithClassificationContextAllRequest.xml
- RecordQueryMetaDataWithClassificationContextAllResponse.xml

**Record Query MetaData - Outputs**

The response contains a list of all the attribute groups the user has access to. It also contains the list of attributes belonging to the attribute groups.

**AttributeGroup**

The data for each attribute group that is part of the repository is returned as an ‘AttributeGroup’ entity which is related to ‘MasterCatalogRecord’ entity.

**RecordAttribute**

The data for each record attribute is returned as a ‘RecordAttribute’ entity that is related to the ‘MasterCatalogRecord’ entity.

Apart from providing record attribute details like name, description, data type etc., the service also shows the attribute group to which the attribute belongs.

**Display Dependency Between Attributes**

The characteristics of an attribute or an attribute’s displayed value depends on the values of another attribute and is returned as ‘DependentRecordAttribute’ relationship.
Role-based Access Privileges

Users and role based privileges can be defined in the system, for access to attribute groups. The behavior is similar to the one described in the previous section.
Record Query ValidValues (Init) Service

Use this web service for the following:

- To list of initial and valid values for all attributes as defined by the rulebase and specified for the repository.
- To provide access privileges configured for attributes.
- To list the list of valid relationship types available for a given record.

The most common use case of this web service is to query the value of a drop-down attribute, which is dependent on the value of another drop-down attribute. This service works on attribute values supplied by the request.

Response XML Structure of Service

The Response XML structure of the Record Query ValidValues service supports the relationships across catalogs. To switch to a backward compatible response structure, set the Backward compatible Web Service Reponses property in the Configurator to true. Setting this property to true enables backward compatible responses for catalogs, which do not have relationships across catalogs. ValidValues response for catalogs having relationships across catalogs will always have a new response structure irrespective of the value of the property. The default value of this property is false.

ValidValuesInit versus ValidValues

- ValidValuesInit — This command qualifier is used when the user wants to fetch the default initial values for a new record being added.
  
  ValidValuesInit executes the new record initialization rulebase followed by the validations rulebase.

- ValidValues — The user after obtaining the values from ValidValuesInit might change some attributes and might want to obtain the valid values for this changed set of attribute values. In this case the ValidValues command qualifier needs to be used.
  
  ValidValues executes the validations rulebase only.
Record Query/ValidValues Service - Inputs

**Command type**

The Command type needs to be set to ‘Query’, and the entity command qualifier is set to ‘ValidValues’.

```xml
<Command type="Query">
    <MasterCatalogRecord commandqualifier="ValidValues" etype="Entity" type="MasterCatalogRecord">
```

For ValidValuesInit, the command qualifier must be ‘ValidValuesInit’.

**Query Directives**

None.

**Support for Bundles**

A Record Bundle is a list of records related to each other by defined relationships. The relationships are defined in the <Relationship> section of the request/response XML. The ‘ValidValues’ web service works on a record bundle. However, ‘ValidValuesInit’ works only with a single repository record.

**External Keys**

The external keys have the following semantics:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTERCATALOGNAME</td>
<td>The name of the repository which contains the records.</td>
<td>Mandatory.</td>
</tr>
<tr>
<td>PRODUCTID</td>
<td>PRODUCTID of the record for which the valid values are sought.</td>
<td>Optional. When this is empty, the system assumes that the record doesn’t exist and is being newly created. In a record bundle, only one record item can have its PRODUCTID empty.</td>
</tr>
<tr>
<td>PRODUCTIDEXT</td>
<td>This is an optional key used to identify a record.</td>
<td>Optional.</td>
</tr>
</tbody>
</table>
**Entity Data**

The Entity data section contains attribute values corresponding to a record. However, in case of 'ValidValuesInit', this section can be empty when a record is being added or values are being queried for the first time.

In case of 'ValidValues', the request needs to include values of the attributes. Given attribute values are used for evaluating the valid values for each attribute. When a record is part of a record bundle, to get valid values as dictated by the rulebase, the request must provide the entity data of all the record items that are part of the record bundle.

**Record Action Intended**

You can specify the action that you want to perform on the data that is being queried. The valid record actions are:

- ADD
- EDIT
- COPY
- VIEW
- MASS_UPDATE
- RECORD_SEARCH

You can specify the record action in the Context section of the request. For example,

```xml
<Context>
  <Variable name="RECORD_ACTION" type="string">ADD</Variable>
</Context>
```

For more details on RECORD_ACTION, refer to Appendix A Context Variables in *TIBCO MDM Studio Rulebase Designer User’s Guide*.

**Record Query/ValidValues Service - Outputs**

**RecordAttribute**

The data for each record attribute is returned as a ‘RecordAttribute’ entity, that is related to the ‘MasterCatalogRecord’ entity.

**RelationshipAttribute**

The data for each relationship attribute for specified Relationship is returned as a 'RelationshipAttribute' entity.

The following data is returned for each attribute, RecordAttribute or RelationshipAttribute.
- Initial value for an attribute
- Access modifier for an attribute.
- List of valid values configured for an attribute.

**Selected or Initial Value for an Attribute**

The ‘Values’ element inside the ‘RecordAttribute’ or ‘RelationshipAttribute’ is used to return the initial value, per rulebase configurations, for a new record.

The value is not returned in cases where access is denied to an attribute, that is, when ‘Visibility’ is set to ‘HIDE’.

**Visibility of an Attribute**

The ‘Visibility’ element inside the ‘RecordAttribute’ or ‘RelationshipAttribute’ denotes the kind of access the user of the service has to the attribute. The possible values are ‘HIDE’, ‘MODIFY’ and ‘VIEW’.

**Valid Values of an Attribute**

The ‘ValidValues’ element inside the ‘RecordAttribute’ or ‘RelationshipAttribute’ contains a list of valid values configured for an attribute. In most cases, this list is used to display a drop-down list.

A sample of valid values response for relationship attribute-

```xml
<Relationship>
  <RelationType>Contains</RelationType>
  <RelatedEntities>
    <MasterCatalogRecord etype="Entity">
      <ExternalKeys>
        <Key name="MASTERCATALOGNAME" type="string">TEST_VALIDVALUES</Key>
        <Key name="PRODUCTID" type="string">bchild</Key>
        <Key name="PRODUCTIDEXT" type="string">bchild</Key>
      </ExternalKeys>
      <RelationshipData>
        <Relationship>
          <RelationType>RelationshipAttributeValidValues</RelationType>
          <RelatedEntities>
            <RecordAttribute etype="Entity">
              <ExternalKeys>
                <Key name="name" type="string">SIZE</Key>
              </ExternalKeys>
              <EntityData xsi:type="RecordAttributeEntityDataType">
                <Value type="string">L</Value>
                <Visibility type="string">MODIFY</Visibility>
                <ValidValues type="map">
                  ...
                </ValidValues>
              </EntityData>
            </RecordAttribute>
          </RelatedEntities>
        </Relationship>
      </RelationshipData>
    </MasterCatalogRecord>
  </RelatedEntities>
</Relationship>
```
Multi-Value Drop Downs Values of a Record Attribute

Some of the attributes in a record might be represented as multi-value drop-down elements. The lists of valid values for these elements are represented as ‘ValidValues’, as described in the following format. Note the ‘type’ attribute is set to ‘marray’ (different from ‘map’ from a single value drop-down described in the previous section).

The first ‘Value’ element in an ‘Entry’ represents the value to be saved or sent to the server. Each of the values after the first one is a column in the multi-value drop down.

Available List of Valid Relationship Types

The list of valid relationship types for a given record are modeled as a ‘RecordAttribute’ with external key ‘name’=‘RELATIONSHIP_LIST’. The ‘Value’ element in the RecordAttribute provides the default relationship type for a given record.

The service also shows the record attribute on which the relationship list is dependent.
Work Item Reference

When a queried record is part of a work item, the response contains the context variable ‘WORKITEMREFERENCE’ that references the work item.

```xml
<Context>
  <Variable name="WORKITEMREFERENCE" type="string">10001</Variable>
</Context>
```

Work item reference is usually used when the Record Validate Service is called.

Configuration Parameters

The following configuration parameters can change the behavior of the Record Query/ValidValues service.

Rulebase Configuration

- **Attribute Groups Hide/Show access** – Based on the rules defined, certain attribute groups can be hidden from the user. The attribute values belonging to the hidden groups are not returned as part of the response; the ‘Visibility’ is marked as ‘HIDE’.

- **Attribute Hide/Show access** – Based on the rules defined, certain attributes can be hidden from the user. These values of hidden attributes are not returned as part of the response; the ‘Visibility’ is marked as ‘HIDE’.

- **Relationship Types Hide/Show access** – Relationships can be marked as ‘hidden’ using the rulebase. Hidden relationships are not returned as part of the response.

- **Propagation** – When a new record is added in an existing bundle, certain attribute values can be inherited from different record items (parent or child) in the record bundle.

- **Initial value assignment** – Initial values can be assigned to certain attributes when a record is being added.

Role-based Access Privileges

Users and role based privileges can be defined in the system, for access to the attribute groups. These privileges are reflected in the ‘Visibility’ attribute, returned as part of the response.
Record Validate Service

Use this web service to validate the attribute values of a record against the validation rules defined in the system. This service works on a record bundle, and can take a list of related records for processing.

The service results in a response being generated with errors, warning and information messages generated by validating the record bundle against the rules defined.

Record Validate Service - Inputs

Command type

The Command type needs to be set to ‘Validate’ and the entity command qualifier is set to either ‘Add’ or ‘Modify’ – depending on whether a record bundle is being added or modified. Note that the qualifier corresponds to the state of the root record (identified by the first MasterCatalogRecord element within a Command) in a bundle. Therefore, if you edit an existing record and add a new record as child to it, the qualifier will still be ‘Modify’ and not ‘Add’.

<Command type="Validate">
<MasterCatalogRecord commandqualifier="Add" etype="Entity" >

Query Directives

None.

Support for Bundles

This web service can work on more than one record as part of a record bundle.

External Keys

Each record in the record bundle must have external keys. The external keys have the following semantics:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTERCATALOGNAME</td>
<td>The name of the repository which contains the records.</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>
Entity Data

Entity data section contains the attribute values corresponding to a record.

Work Item Reference

When a record is validated in the context of a work item, the request must contain the context variable ‘WORKITEMREFERENCE’ that references the work item.

```
<Context>
  <Variable name="WORKITEMREFERENCE" type="string">10001</Variable>
</Context>
```

Work item reference is usually obtained from the Record Query/ValidValues service.

Record Validate Service - Outputs

Records in the record bundle are validated against validation rules defined on the server, and the results are returned in the response.

ResultList

For more information, refer to, ResultList, on page 7.

TransactionResult

For more information, refer to, TransactionResult, on page 7.

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODUCTID</td>
<td>PRODUCTID of the record for which the valid values are sought.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>PRODUCTIDEXT</td>
<td>This is an optional key used to identify a record.</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Record Add or Modify Services

This web service allows you to add or modify a record in a repository. This service works on record bundles, and can therefore take a list of records for processing. In case the records have errors, the service returns a response that contains details of the validation errors (as described in the previous section).

If the records are successfully added or modified, an informational message is sent indicating whether the records were saved and if the workflow was initiated.

Flexible Editing of Records

Some features that provide flexibility during adding or modifying a record are:

- Defining a relationship — The entire set of child records is not required to define a relationship between two records; only a related record can be specified. To save a record with the relationship, you need not specify the entire set of the record; specifying the relationship data is sufficient.

- Specifying a reverse relationship — You can also specify and process a reverse relationship.

- Adding or Modifying Records Incrementally — You can add and modify records incrementally. Incremental handling is controlled by the `<Relationship incremental="true" />` flag. By default, the flag is set to `true`. When the flag is set to `true`, only the record relationship is added to the given existing record. If the incremental flag is set to `false`, all the other relationships are deleted and the newly specified relationship is added.

For example, let us say, A-Contains-B and the Incremental flag is set to true. If a request is sent as A-Contains-C, A will be related to B,C. If the Incremental flag is set to false, and the request is sent A-Contains-C, the relationship A-Contains-B is deleted and A will have a new relationship with C.

- Deleting a Relationship Between Records — You can delete relationships between records by specifying the `<Relationship delete="true" />` flag.

For example, let us say, A-Contains-B,C.

— To delete a specific relationship with the record, send a request A-Contains-B. In the relationship element of this request, set the
<Relationship delete="true">. Now "A" will be related to C. Relationship between A and B is deleted.

— To delete all relationships with records for a given relationship, send a request to modify a record where <Relationship delete="true">, specify the relationship type, and do not specify any record details.

— To delete relationships with records for all relationships, send a request to modify a record, specify <Relationship delete="true" incremental="false">, and specify (*) as relationship type. (*) indicates all relationships.

**Saving a Record as DRAFT**

You can save a record as DRAFT and initiate a workflow for that record. To save the record as DRAFT, specify the record_state as DRAFT. A workflow will be initiated.

- The supported command types for saving a record as DRAFT are 'add' and 'modify'.
- The entity command qualifier supported is validate/process and novalidate/process.

The command qualifiers NoValidate/NoProcess and Validate/NoProcess are not supported when saving the record as DRAFT. If any of these two qualifiers are specified, an error is returned.

- Flexible editing of records is supported while saving a record as DRAFT.

**Modifying a Record as DRAFT**

When saving a record as DRAFT in the context of a workflow, new workflow is not triggered but the record is modified and only the event state is updated.

- If a work item reference is included, the modification is a part of the corresponding workflow and new work item is not generated.
- If a work item reference is not included, DRAFT is saved and new workflow is started.

**Future Effective Date (FED) Records**

For Future Effective Date records, you must define an ID, Extension, and Version. Additionally, you can add the following

- Add multiple versions of FED records.
Create a new version of record, if you have made changes to the record data and/or to the EffectiveDate.

The EffectiveDate specified at the context level applies to all records and relationship. However, to overwrite it on individual record level, specify the new data value for the EFFECTIVEDATE attribute.

For a relationship with EffectiveDate, specify the relationship effective date in context so that related record is returned according to the relationship effective date.

While modifying a FED record version, the RECORD_VERSION parameter is mandatory. For example,

```xml
<Key name="RECORD_VERSION" type="int">1</key>
```

In the Record Add web service, if you do not specify EFFECTIVEDATE, the record is saved as a current timestamp record and a confirmed version is created.

### Classifying Records

You can classify records into the specified classification using the `<ClassificationSchemes>` context while adding or modifying a record.

You can classify records only for classification schemes for which the Manual extraction method is selected.

The following actions are supported for the classification scheme:

- **Classify** – Classifies a record into the specified classifications. This is default action.
- **Unclassify** – Unclassifies a record from the specified classifications.
- **UnclassifyAll** – Unclassifies a record from all existing classifications.
- **Replace** – Unclassifies a record from all existing classifications and classifies them into specified classifications.

Specify the classifications in the following sequence:

- Name of a classification scheme
- Actions for a classification scheme
- Classification code: Specify the code hierarchy by nesting the `ClassificationCode` tag. Ensure that you specify the classification code and not the classification code name.

Refer to the following request example:

```xml
<ClassificationSchemes>
```

```xml
</ClassificationSchemes>
```
<ClassificationScheme name = "GamesAndToys" action="Classify">
  <ClassificationCode code="86010100">
    <ClassificationCode code="10005133"/>
    <ClassificationCode code="20000045"/></ClassificationCode>
  </ClassificationCode>
</ClassificationScheme>

Specify the exact hierarchy of classifications in the context starting from level one. If there is a mismatch in hierarchy, an error is returned.

Alternatively, instead of classifying record by adding it in the request you can add it in the context variable.

<Context>
  <Variable name="CLASSIFICATIONCODEPATH">
   GamesAndToys/86010100/10005133/20000045</Variable>
</Context>

and have the following rulebase

<constraint>
  <name>rulebase-ACTION-CATEGORIZE</name>
  <action>
    <categorize incremental="true">
      <var>context_classificationcodepath</var>
    </categorize>
  </action>
</constraint>

CLASSIFICATIONCODEPATH in webservice context is a Name- Value pair. You can use any Name-Value pair to do classification using rulebase and webservice context. SchemeNames followed by a forward slash (/) delimeter is used to define a complete codepath. If you have multiple codepath use comma(,) separator.

You can use it with Record ADD/MODIFY, Record Query web services. It can be used to get List of Values, provide access based on the category.

The sample request and response XML files for all actions are available in the following locations:

- Classify Action:
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Add-ManualClassify-Request.xml
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Add-ManualClassify-Response.xml
• Unclassify Action:
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Modify-ManualUnclassify-Request.xml
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Modify-ManualUnclassify-Response.xml

• Unclassify All Action:
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Modify-ManualUnclassifyAll-Request.xml
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Modify-ManualUnclassifyAll-Response.xml

• Replace Action:
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Modify-ManualReplace-Request.xml
  - $MQ_HOME/schema/DataService/2.0/samples/repository/record-add_modify_delete/MasterCatalogRecord-Modify-ManualReplace-Response.xml

Record Add/Modify Services - Inputs

Command type
The valid command types are:

• **Add** – Use when the root record in a bundle is being added.
• **Modify** – Use when the root record in a bundle is being modified.

Set the entity command qualifier to any of the following:

• **Validate/Process** – Causes the records to be validated before saving. This also initiates New Record Introduction Edit (NRIE) workflows. This is the default.
• **Validate/NoProcess** – Validates records and save them without starting the NRIE workflows.
• **NoValidate/Process** – Does not validate the records, but saves them, and also starts NRIE workflows.
• **NoValidate/NoProcess** – Does not validate the records or start the workflow, but saves them.
• **NoValidate/ProcessOnDataChange (only for the Modify command type)** – Does not validate records, however processes and saves the records when the
data has changed for the specified records. If the data has not changed, an error is returned.

- **Validate/ProcessOnDataChange (only for the Modify command type)** – Validates the records, however processes and saves the records only when the data has changed for the specified records. If the data has not changed, an error is returned.

**Execution Mode**

Set the execmode to either `SYNCHR` or `ASYNCHR`. For more information, refer to [Execution Mode on page 2](#).

**Handling File Type Attributes**

TIBCO MDM can handle file type attributes either as attachments or as relative paths into the application. To perform this, specify the `FILE_TYPE_ATTRIBUTES` context variable with the following two valid values:

- **ATTACHMENT** – Indicates that the client has sent the file type attributes as SOAP attachments.

- **HREF** – Indicates that the client had already loaded the files onto the server (using HTTP post), and wants the server file type attributes to be processed as relative paths into the application. This is the default.

```xml
<Context>
  <Variable name="FILE_TYPE_ATTRIBUTES" type="string">ATTACHMENT</Attribute>
</Context>
```

**Handling System Generated Attributes**

You cannot change the following system-generated attributes. If any of the following attributes are specified in the request XML, an error is returned.

- `Active`, `creation date`, `moddate`, `version number`, `product keyID`, `modmemberID`, `checksum`, `importtime stamp`, `ownerID`, `ownerType`, `lastConfirmedVersion`, `processLogID`, `batchID`, `parentVersion`, `catalogVersionNumber`

The `CREATIONDATE` and `MODDATE` system attributes do not provide the time zone details. This is a limitation. For example, the value for these attributes do not display the time zone ID along with the date and time, that is, `2013-05-08 03:19:46.0 Asia/Shanghai`. 
Query Directives

None.

Support for Bundles

This web service can work on more than one record, as part of a record bundle.

External Keys

Each record in the record bundle must be external keys. The external keys have the following semantics:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTERCATALOGNAME</td>
<td>The name of the repository which contains the records.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>PRODUCTID</td>
<td>PRODUCTID of the record for which the valid values are sought.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>PRODUCTIDEXT</td>
<td>This is an optional key used to identify a record.</td>
<td>Optional</td>
</tr>
</tbody>
</table>

Entity Data

Entity data section contains the attribute values corresponding to a record.

RECORD_STATE

The request can provide the state in which the records need to be saved. The state is accepted as an attribute in the entity data with the name RECORD_STATE. The valid states are:

- **UNCONFIRMED**: If the Validate/NoProcess command type is specified, by default the UNCONFIRMED state is considered.
- **CONFIRMED**: If the Validate/Process command type is specified, by default the CONFIRMED state is considered.
- **DRAFT**

If an attribute is a multi-value attribute, it is represented as follows:

```xml
<MultiValueAttribute type="string" name="COLOR">
  <Value>red</Value>
  <Value>green</Value>
  <Value>yellow</Value>
</MultiValueAttribute>
```
In this example, COLOR is a multi-value attribute.

If an attribute is defined as a relationship attribute, it is represented as follows:

```xml
<Relationship>
  <RelationType>Contains</RelationType>
  <RelatedEntities>
    <MasterCatalogRecord etype="Entity">
      <ExternalKeys>
        <Key name="MASTERCATALOGNAME" type="string">Customer1</Key>
        <Key name="PRODUCTID" type="string">a2-demo1TEST3</Key>
        <Key name="PRODUCTIDEXT" type="string">a2-demo1TEST3</Key>
      </ExternalKeys>
      <EntityData>
        <Attribute name="QUANTITY" type="integer">1</Attribute>
      </EntityData>
    </MasterCatalogRecord>
  </RelatedEntities>
</Relationship>
```

In this example, QUANTITY is a relationship attribute.

For a sample of multi-value attributes and relationship attributes, refer to $MQ_HOME/schema/DataService/2.0/samples.

**WorkItem Reference**

When a record is modified in the context of a work item, the request must contain the context variable ‘WORKITEMREFERENCE’ that references the work item before the `<Transaction>` element.

```xml
<Context>
  <Variable name="WORKITEMREFERENCE" type="string">10001</Variable>
</Context>
```

The work item reference is usually obtained from the Record Query/ValidValues or Work Item Query service.

**Record Add/Modify Services - Outputs**

Based on the command type and the command qualifier, the system tries to validate and save the records. Any errors found during validation (if asked to be performed), are returned in the response.

If validation is attempted and results in errors, the records are not saved.

**ResultList**

For more information on what the ResultList element contains, refer to the section, ResultList, on page 7.
**Transaction Result**

For more information, refer to *TransactionResult*, on page 7.
Record Delete Service

This web service allows you to delete:

- A repository record
- Records related to specified record
- Both the repository record and related records. That is, delete the parent record and the records for the given relationship.
- A future dated record version

When a record is deleted, a workflow is initiated unless specified otherwise in command qualifier. Using the right command qualifier, you can skip the workflow processing and complete the delete record operation.

If the workflow is initiated its DOCSUBTYPE is set as CatEditDelete.

The eligibility criteria for deletion is:

- You must have access permission to delete the record on the specified repository.
- The record must not be a part of another workflow. If the record is a part of another workflow, the record is not deleted.
- When deleting record and related records, all the records must be available for deletion. If deletion is not allowed on one of the records then the whole transaction is disallowed and no deletion will be carried for the remaining records even if deletion is allowed. Deletion will be successful only if all the records are available for deletion. The record is not available for deletion when the record is being processed in other workflows or the user does not have the permission for deletion.

Deleting Record

When a record is deleted, the latest version of the record must be:

- Active
- The record is not in another workflow

The delete is performed by creating a new version with Active = N.
**Future Effective Date (FED)**

The web service can delete a specific future effective dated version if
- version number is specified
- the specified version is future dated and active
- the version is not in another workflow
- delete the previous FED versions by specifying DELETE_PREVIOUS_FED_VERSION in the context. Alternatively, you can retain the previous FED record version or overwrite it. Later, delete it when the EffectiveDate is changed for a record.

The delete is performed by marking the future dated version as REJECTED.

For example:

```xml
<Command type="Delete">
  <MasterCatalogRecord etype="Entity"
    commandqualifier="Record">
    <ExternalKeys>
      <Key name="MASTERCATALOGNAME" type="string">MASTERCATALOG_WITH_EFFECTIVE
        DATE_RA</Key>
      <Key name="PRODUCTID" type="string">EffectiveDate</Key>
      <Key name="PRODUCTIDEXT" type="string">EffectiveDate</Key>
      <Key name="RECORD_VERSION" type="int">2</Key>
    </ExternalKeys>
    <EntityData>
      <!--default RECORD_STATE is if not specified is UNCONFIRMED-->
      <Attribute name="EFFECTIVEDATE" type="date">2011-06-29
        00:00:00.000</Attribute>
    </EntityData>
  </MasterCatalogRecord>
</Command>
```

**Delete Record - Inputs**

**Command Type**

The valid command types are:
- ‘Delete’ – This should be used when you need to delete a repository record or related target records.

The entity command qualifier is set to any of the following:
- ‘Record’ – Deletes the specified record and initiates the delete workflow.
- ‘RelationshipTargets’ – Deletes only the related records and initiates the delete workflow. Optionally, relationship name can be specified.
• ‘Record/NoProcess’ – Deletes the specified record and confirms it immediately without initiating the workflow. User must have required privileges.

• ‘RelationshipTargets/NoProcess’ – Deletes only the related records and confirms it immediately without initiating the workflow. Optionally, relationship name can be specified. User must have required privileges.

• ‘DeleteAll’ – Deletes the specified record/related records and initiates the workflow. Optionally, relationship name can be specified.

• ‘DeleteAll/NoProcess’ – Deletes the specified record/related records and confirms the record immediately. Optionally, relationship name can be specified. User must have required privileges.

• Command qualifier not specified or blank – This causes both the record and the relationship target records to be deleted and delete workflow is initiated. It is recommended that DeleteAll be used instead of a blank command qualifier.

**Execution Mode**

The execmode can be set to either SYNCHR or ASYNCHR. For more information, refer to , Execution Mode, on page 2.

**Query Directives**

None.

**External Keys**

To delete a record you must specify either of the following:

- productkeyid
- productid AND productidext

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODUCTKEYID</td>
<td>The PRODUCTKEYID value of the record. Either specify productkeyid or productid and productidext.</td>
<td>Either one of these keys is mandatory.</td>
</tr>
<tr>
<td>PRODUCTID (and) PRODUCTIDEXT</td>
<td>PRODUCTID and PRODUCTIDEXT of the record for which the valid values are sought. Either specify productkeyid or productid and productidext.</td>
<td></td>
</tr>
</tbody>
</table>
Delete Record - Outputs

Based on the command type and the command qualifier, the system tries to delete records. Any errors found during validation (if asked to be performed), are returned in the response.

**ResultList**

For more information, refer to, *ResultList, on page 7.*

**TransactionResult**

For more information, refer to, *TransactionResult, on page 7.*
Initiate Workflow Service

This web service is used to initiate workflow for a given record.

You must specify the command type as 'InitiateWorkflow' and the entity command qualifier as 'docSubType' to invoke a workflow for a record. Before initiating a workflow for a docsubtype, make sure that the docsubtype entry is in the Process Definition Selection business process. Triggering a workflow is an asynchronous process, so the response is returned immediately. It does not wait for completion of the workflow. Because of this you will always see a success message in the response message.

Records with RECORD_STATE as draft cannot be used to initiate a workflow.

Initiate Workflow Service - Inputs

Command type
A valid Command type is: ‘InitiateWorkflow’. This should be used when you need to initiate a workflow.

Command qualifier
The entity command qualifier is set to the following:
‘docSubType’ – Specify an existing docSubType. For example, FACTSHEET, CatEditChange, CatEditDelete, CatEditImport, CatEditAdd, and CatEditAccepted.

<Command type="InitiateWorkflow">
<MasterCatalogRecord commandqualifier="CatEditAdd" etype="Entity" />

Execution Mode
The execmode can be set to either SYNCHR or ASYNCHR. For more information, refer to, Execution Mode, on page 2.

Query Directives
None.
Support for Bundles

None.

External Keys

Each web service request must have external keys. The external keys have the following semantics:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTERCATALOGNAME</td>
<td>The name of the repository which contains the records.</td>
<td>Mandatory.</td>
</tr>
<tr>
<td>PRODUCTID</td>
<td>PRODUCTID of the record.</td>
<td>Mandatory.</td>
</tr>
<tr>
<td>PRODUCTIDEXT</td>
<td>PRODUCTIDEXT of the record.</td>
<td>Mandatory.</td>
</tr>
<tr>
<td>RECORD_STATE</td>
<td>This key identifies the state of the record being searched. Valid values are ‘UNCONFIRMED’ (the default) or ‘CONFIRMED’.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>• If the RECORD_STATE is set to ‘CONFIRMED’, the system looks for the latest ‘CONFIRMED’ record.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• If state is set to ‘UNCONFIRMED’, the system looks for the latest record including ‘UNCONFIRMED’ records, that is, if the latest record is an unconfirmed one, that would be returned. This is the default behavior, so if unconfirmed exists select it, otherwise return the last confirmed version.</td>
<td></td>
</tr>
</tbody>
</table>

Initiate Workflow - Outputs

**ResultList**

For more information, refer to, ResultList, on page 7.
Transaction Result

For more information, refer to, TransactionResult, on page 7.
**Transaction Atomicity for Record Web Services**

Use *Transaction Atomicity* to atomize the transaction on the commands included in the `<Transaction>` element. If a command specified in the `<Transaction>` element fails, the transaction is not processed. Transaction atomicity applies to the Create, Update, and Delete operations performed on a record. For more information on these operations, refer to **CRUD Operations, page 56**.

The Transaction Atomicity feature currently does not support synchronous workflows due to concerns over long running or suspended workflow logic.

To enable the Transaction Atomicity feature, specify the `atomic` attribute in the `<Transaction>` element. You can specify any one of the following values for the `atomic` attribute:

- **on**: all commands are processed together as an atomic operation. You can set `atomic="on"` for the following web services:
  - Record Add or Modify Services, page 96
  - Record Delete Service, page 105

- **off**: each command is processed as a separate transaction.

The Transaction Atomicity feature does not work for the Record Query web service. If you specify `atomic="on"` for the Command type="Query", the following error message is displayed:

*Only Record Add/Modify/Delete commands should be part of atomic transaction.*

The Add and Modify commands of the same record are not supported in a atomic transaction because the record should be persisted before to get it modified.

For non-atomic transaction, when a request contains two related MasterCatalogRecords in one command, specify the commands in the following order:

1. Add parent record
2. Add child record
3. Modify parent record to establish relationship between parent and child records
Using the following scenarios, you can process a record web service request with the multiple commands performing CUD operations. The operations are specified as an atomic process within the <Transaction> element.

- Scenario 1: Process Multiple CUD Commands, page 114
- Scenario 2: Large Number of Modified Records, page 115
- Scenario 3: Enable Global Distributed Locking, page 115
Scenario 1: Process Multiple CUD Commands

In this scenario, you can add or modify records with multiple commands in a single transaction. Specify `atomic="on"` in the `<Transaction>` element. You can also add multiple transactions with multiple commands.

Example

```xml
<Transaction atomic="on">
  <Command type="Modify">
    <!--The Supported CommandTypes for Save are Add and Modify-->
    <MasterCatalogRecord etype="Entity" commandqualifier="Validate/Process">
      <!-- The Suppored commandqualifier for Save are Validate/Process, Validate/NoProcess, NoValidate/Process, NoValidate/NoProcess default is Validate/Process-->
      <ExternalKeys>
        <Key name="MASTERCATALOGNAME" type="string">Customer</Key>
        <Key name="PRODUCTID" type="string">1</Key>
        <Key name="PRODUCTIDEXT" type="string">1-Ext</Key>
      </ExternalKeys>
      <EntityData>
        <Attribute name="PRODUCTIDEXT" type="string">1</Attribute>
        <Attribute name="PRODUCTID" type="string">1-Ext</Attribute>
        <Attribute name="SHORTDESC" type="string">Record for Customer repository</Attribute>
      </EntityData>
    </MasterCatalogRecord>
  </Command>
</Transaction>
```
Result

All the commands are processed as a single atomic operation. Records are successfully modified. If any of the commands in a transaction fails, the transaction is not processed and the records are not added or modified. The transaction may fail because the non-supported command qualifier or an erroneous rulebase exists in a command.

Scenario 2: Large Number of Modified Records

In this scenario, you can process multiple CUD commands with the sizeable amount of records and relationships. The Transaction Atomicity feature enforces consistency between a set of single data changes. Use the Transaction Atomicity feature to load multiple unrelated records, that is, bulk load. However, the number of records should not be large, in particular, if these are bundles. For a single data set, do not load more than 30-50 records in one transaction.

For example, to create a large bundle with a depth of four levels, specify 4 as the value for the Validation Record Bundle Depth property in Configurator. The property defines the depth for validations of related records in the bundle. Add propagation rulebases to affect numerous records. Specify atomic="on" in the <Transaction> element.

Scenario 3: Enable Global Distributed Locking

In this scenario, concurrent record web service requests are sent with the multiple commands in a single transaction. The entire transaction is locked instead of a separate command specified in the transaction.

1. Before executing this operation, specify true for the Enable Save Lock property in Configurator. The property enables the locking process.

2. Specify atomic="on" in the <Transaction> element to lock the entire transaction.
Chapter 3  Work Item Services

This chapter describes the usage and customization of work item services.

Topics

- Work Item Query Parameters Service, page 118
- Work Item Query Service, page 120
- Work Item Modify Close Service, page 126
- Work Item Lock Service, page 130
- Work Item UnLock Service, page 132
- Get Work Item Service, page 134
- Work Item Reassign Service, page 140
- Get Custom Work Item Summary Service, page 142
- Get Predefined Work Item Summary Service, page 148
Work Item Query Parameters Service

This web service can be used to get a list of query parameters supported by the work item query service. This service also provides valid values, if any, for each of the supported parameters.

Work Item Query Parameters Service - Inputs

Command type

Command type needs to be set to ‘Query’ and the entity command qualifier needs to be ‘Parameters’.

<Command type="Query">
<Workitem commandqualifier="Parameters" etype="Entity">

Query Directives

None.

External Keys

None.

Work Item Query/Parameters Service - Outputs

The list of query parameters for work lists supported by the system is returned by the service.

Entity Data

- **FromDate** — The start date from which the work list is required.
- **ToDate** — The end date until which the work list is required.
- **RecordAttribute** — Value of a record attribute.
- **Eventreference** — The ID of the event for which work items need to be searched.
- **ShowActionsFirst** — Specifies whether action items should be shown first.
- **PRODUCTID** — ID of the record.
- **PRODUCTIDEXT** — Extension of the record. Optionally used to identify a record.
- **States** — The state of the work items.
- **Datapools** — Name of the datapool involved in the workflow.
- **Partners** — Name of the partner involved in the workflow.
- **WorkitemTypes** — Type of work item. Valid values are "Notifications" and "Action Items".
- **DocumentTypes** — The document type associated with the work item.
- **DocumentSubTypes** — The document sub type associated with the work item.
- **Owners** — The username of the owner of the work item.
- **MasterCatalogNames** — The name of the repository which contains the records.

In case the query does not have a valid value list, it is represented as an empty attribute. For example:

```xml
<Attribute name="ToDate" type="date"/>
```
Work Item Query Service

This web service can be used to query work items, based on the search criteria specified.

Work Item Query Service - Inputs

Command type

Command type needs to be set to ‘Query’ and the command qualifier must be empty.

<Command type="Query">
<Workitem etype="Entity">

Query Directives

<MaxCount> can be specified to limit the number of work items returned, should more than one work item match the query.

<StartCount> can be specified to achieve windowing/pagination of a large result set. In combination with MaxCount, the caller can control which ‘page’ of the result set needs to be returned.

<Command type="Query">
<StartCount>30</StartCount>
<MaxCount>10</MaxCount>

External Keys

The external keys are used to locate list of parameters and their valid values.

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATE</td>
<td>The state of the work items.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Default is ‘OPEN’.</td>
<td></td>
</tr>
<tr>
<td>FROMDATE</td>
<td>The start date from which the work list is required.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Default is null (no start date).</td>
<td></td>
</tr>
<tr>
<td>TODATE</td>
<td>The end date until which the work list is required.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Default is current date.</td>
<td></td>
</tr>
<tr>
<td>Key</td>
<td>Meaning</td>
<td>Required</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>WORKITEMTYPE</td>
<td>Type of work items being queried.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Valid values are: Action Items, Notifications, All; Default ‘ALL’.</td>
<td></td>
</tr>
<tr>
<td>DOCUMENTSUBTYPE</td>
<td>The document sub type.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Default ‘ALL’.</td>
<td></td>
</tr>
<tr>
<td>DATAPool</td>
<td>The name of the datapool or integration hub.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Default ‘ALL’.</td>
<td></td>
</tr>
<tr>
<td>PARTNER</td>
<td>The name of the partner or backend system.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Default ‘ALL’.</td>
<td></td>
</tr>
<tr>
<td>SHOWACTIONSFIRST</td>
<td>Whether action items should be shown first.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>Default – worklist is unsorted.</td>
<td></td>
</tr>
<tr>
<td>OWNER</td>
<td>The username of the owner of the work item. If the owner is not the same</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td>as the user who initiated the web service, the owner must have permissions</td>
<td>Default – The username of the</td>
</tr>
<tr>
<td></td>
<td>to view the work items of other users.</td>
<td>profile used in the service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request.</td>
</tr>
<tr>
<td>RECORDATTRIBUTE</td>
<td>Value of a record attribute.</td>
<td>Optional.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> For query by this parameter to work, configurations are</td>
<td></td>
</tr>
<tr>
<td></td>
<td>necessary as part of the inbox customization.</td>
<td></td>
</tr>
<tr>
<td>EVENTREFERENCE</td>
<td>Search work items for the specified event.</td>
<td>Optional.</td>
</tr>
<tr>
<td>Key</td>
<td>Meaning</td>
<td>Required</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>CUSTOMATTRIBUTE</td>
<td>Search work items based on custom attributes. For example, <code>&lt;Key name=&quot;CustomAttribute&quot;&gt;LegacySystem&lt;/Key&gt;</code></td>
<td>Optional. Default is null</td>
</tr>
</tbody>
</table>
| NATIVECONDITION       | Search work items based on specified search conditions. You can specify this key along with other conditions. The Search condition must be specified in the following formats:  
  - Native SQL format valid for configured database  
  - Only the WHERE clause (Do not include values to select.)  
  - The WHERE clause must assume a join on the WORKITEM and WORKITEMDETAIL tables. Alias for the WORKITEMDETAIL table is workitemdetail and alias for the WORKITEM table is workitem.  
  The condition is applied as AND with other conditions and it is applied without any processing. If it is invalid, an SQL error is thrown. The application does not display the error. For example, `<Key name="NativeCondition">workitemdetail.name = 'PRODUCTVERSION' and workitemdetail.value = '2'</Key>` | Optional. Default is null     |
Work Item Query Service - Outputs

The list of work items fitting the given search criteria (using external keys) are returned in the result.

**Entity Data**

Each work item returned in the result has the following attributes:

- **ActivityName** — Name of the work item activity.
- **Age** — Age of the work item. The format of age is D:H:M:S, which refers to date, hour, month, and seconds.
- **ClosedBy** — Login name of the person who closed the work item. Returned only if workitem is closed.
- **ClosedByUserName** — The first name and last name of the person who closed the work item. Returned only if workitem is closed.
- **ClosedOn** — Date on which the work item was closed. Returned only if workitem is closed.
- **CreationDate** — Date on which workitem was created.
- **Datapool** — Name of the marketplace involved in the workflow.
- **Description** — This is a dynamically generated description based on inbox configuration. This contains tokens to indicate the type of hyperlinks needed as part of the description. Please refer the *Inbox configuration section* for more details.
- **DocumentType** — The document type associated with the work item.
- **DocumentTypeDescription** — The description of the document type in the language of the user.
- **DocumentSubType** — The document sub type associated with the work item.

---

[Table of Key Features]

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
</table>
| KEYWORD     | Search work items based on the tagged keywords. Keywords are configured when a work item is created in the workflow. For example, 
<Key name="Keyword">HighPriority</Key> | Optional. Default is null |

---

Optional. 
Default is null
- **DocumentSubTypeDescription** — The description of the document sub type in the language of the user.

- **Eventreference** — The ID used to reference the event to which a work item belongs.

- **Organization** — Name of the organization to which a work item belongs.

- **OriginalWorkitemID** — Only returned if applicable (for reassigned workitems only).

- **OriginalOwner** — Login name of the original owner of the work item. Returned only if applicable (for reassigned workitems).

- **OriginalOwnerName** — The first name and last name of the original owner of the work item. Returned only if applicable (for reassigned workitems).

- **Owner** — The first name and last name of the owner of the work item. This is useful when the work supervisor wants to look at work items of all users in an organization.

- **OwnerName** — The first name and last name of the owner of the work item.

- **OwnerType** — Role/Member.

- **Partner** — Name of the trading partner involved in the workflow.

Datapool and Partner are output only if they are associated with the work item.

- **State** — State of the work item. OPEN/OPEN_TIMEDOUT/CLOSED and so on.

- **StateDescription** — Description of work item state in user preferred language.

- **TaskType** — The type of workflow. Example, CATALOGMESSAGE.

- **Workitemreference** — A unique reference for each work item.

- **WorkitemType** — Type of work item. Valid values are "Notifications" and "Action Items".

**Records in Work Item**

With each work item, the list of records associated with it are also listed. This is represented as a ‘Relationship’ of type ‘MasterCatalogRecord’.

For more information, refer to Records in Work Item, on page 136 in the Get Work Item Service.
Work Item Searching
You can search the work item based on a specific criteria. For example, you can add tags to work items using the CreateWorkitem activity, and then search for it.

Work Item Locking
For more information, refer to Work Item Lock Service, on page 130.

Transaction Result
For more information, refer to TransactionResult, on page 7.
Work Item Modify Close Service

This web service can be used to close a list of work items.

Work Item Modify/Close Service - Inputs

Command type

- **Close** — Work item is closed. The Command type needs to be set to ‘Modify’, and the entity command qualifier needs to be ‘Close’.

  ```xml
  <Command type="Modify">
  <Workitem commandqualifier="Close" etype="Entity">
  </Workitem>
  </Command>
  ```

- **Close/Approve** — Work item is approved. The approval data can be specified using custom input parameters. If the ‘result’ custom parameter is specified, it is ignored.

  ```xml
  <Command type="Modify">
  <Workitem commandqualifier="Close/Approve" etype="Entity">
  </Workitem>
  </Command>
  ```

- **Close/Reject** — Work item is rejected. The rejected data can be specified using custom input parameters. If the ‘result’ custom parameter is specified, it is ignored.

  ```xml
  <Command type="Modify">
  <Workitem commandqualifier="Close/Reject" etype="Entity">
  </Workitem>
  </Command>
  ```

The work items are closed one by one. If any of the work items cannot be closed, it is ignored. The number of closed work items is reported in the results along with the list of errors.

Besides, the ‘Close’ command qualifier, you can also specify the following command qualifiers:

- **CloseAll** — Closes all work items based on the specified search criteria. All work items selected using the search criterion are considered for closing. Similar to ‘Close’, any inputs to actionable work items can be supplied. All closed work items receive same parameters, such as mass approval or rejections.

  ```xml
  <Command type="Modify">
  <Workitem commandqualifier="CloseAll" etype="Entity">
  </Workitem>
  </Command>
  ```
• **CloseAll/Approve** — Approves all work items based on the specified search criteria except the FormResult custom parameter. If the FormResult custom parameter is specified, it is ignored.

```xml
<Command type="Modify">
  <Workitem commandqualifier="CloseAll/Approve" etype="Entity">
```

• **CloseAll/Reject** — Rejects all work items based on the specified search criteria except the FormResult custom parameter. If the FormResult custom parameter is specified, it is ignored.

```xml
<Command type="Modify">
  <Workitem commandqualifier="CloseAll/Reject" etype="Entity">
```

The `<MaxCount>` refers to the number of closed work items, which limits the total number of closed work items. The default MaxCount is 20. For example,

```xml
<Command type="Modify">
  <MaxCount>10</MaxCount>
  <Workitem commandqualifier="CloseAll" etype="Entity">
```

If Close or CloseAll command qualifier is used, the FormResult parameter is required for actionable work items. As the FormResult parameter is deprecated, do not use Close and CloseAll for actionable work items; instead use Close/Approve, Close/Reject, CloseAll/Approve, or CloseAll/Reject.

**Execution mode**

The execmode can be set to either SYNCHR or ASYNCHR. For more information, refer to Execution Mode, on page 2.

**Query Directives**

NONE.

**External Keys**

The external keys are used to locate a work item.

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKITEMREFERENCE</td>
<td>A reference ID for work item. This is usually available as a result of the work item query service.</td>
<td>Mandatory.</td>
</tr>
</tbody>
</table>
Accept/Reject and Comment

While closing a work item, you can indicate if the work item is accepted or rejected. Specify a comment for the rejection or acceptance. For example:

```xml
<EntityData>
  <Comment etype="Attribute" type="string"> description</Comment>
  <FormResult etype="Attribute" type="string">Reject</FormResult>
  <SendForCorrections etype="Attribute" type="string">N</SendForCorrections>
</EntityData>
```

You can specify a list of work items in the request, and each of them will be closed. However, each work item needs to be in its own transaction, that is, separated by Transaction sections. You can close a work item inspite of errors or rejections. Ensure that close is not called in such situations. You can inquire about errors, warnings, and rejections using the GET WORKITEM web service.

Custom User Inputs

While closing work items, you can specify additional custom inputs. This data is added to the workflow request document and processed using the workflow forms.

Example 1:

Consider a scenario where you need to enter a LaunchDate for a set of records in a work item form. The date can later be used in the workflow. Specify the custom inputs as shown in the following example:

```xml
<EntityData>
  <Comment etype="Attribute" type="string">Reject through Web Service MASS APPROVAL</Comment>
  <FormResult etype="Attribute" type="string">Reject</FormResult>
  <SendForCorrections etype="Attribute" type="string">N</SendForCorrections>
  <!-- The following custom fields are captured as part of the work item form -->
  <LaunchDate eType="Attribute" type="date">2006 10 12</LaunchDate>
</EntityData>
```

The LaunchDate custom input is later included in the output document of the work item activity.
Example 2:

The custom attributes are used to pass in additional information to the application so that the values of these attributes can be used to update the workflow XML document (mlXML). An XPath expression entry is created in the specific UI form on how to access these attributes from the request XML. These extracted values are used to populate the workflow document.

```xml
<EntityData>
  <Comment etype="Attribute" type="string">Reject through Web Service MASS APPROVAL</Comment>
  <FormResult etype="Attribute" type="string">Reject</FormResult>
  <SendForCorrections etype="Attribute" type="string">N</SendForCorrections>
  <!-- The following custom fields are captured as part of the work item form -->
  <raiseAlarm eType="Attribute" type="string">alarm1</raiseAlarm>
</EntityData>
```

- When the `FormResult` value is **Accept**, the value of `SendForCorrections` is not considered.
- When the `FormResult` value is **Reject**, the value of `SendForCorrections` is considered.

Since release 8.2.0 onwards, the `<FormResult etype="Attribute" type="string">Reject</FormResult>` is deprecated.

Specify all custom attributes in the `$MQ_HOME/common/standard/forms/fm26ca.xml` file with the XPath as `//Field/ID[text()=CustomAttribute]/../OutputTo`.

**Work Item Modify/Close Service - Outputs**

The service attempts to close the work item. If any errors occur (example – a non admin trying to close someone else’s work item), they are returned as part of the response.

**ResultList**

For more information, refer to **ResultList, on page 7**.

**Transaction Result**

For more information, refer to **TransactionResult, on page 7**.
Work Item Lock Service

This web service allows you to lock and relock a work item to prevent other users from opening the work item.

**Locking**

A "shared" work item is created when a workflow assigns work items to more than one user in a single work item step. By locking the work item, the user can indicate that he/she has started working on it and other users should wait for him/her to complete or unlock the work item. This applies to scenarios when work items are assigned to a pool of users, all of whom cannot work on the task in parallel.

If a work item is locked by a user, other users can see the work item in the Inbox but cannot open it. A work supervisor, however, can lock the work item for any user. A work supervisor can see the work item in the inbox, open it, and make changes to records within the work item even if it is locked by another user. However, such work items cannot be submitted.

Work item locking only applies to actionable work items, and has no impact on notification work items.

When a work item is locked through a web service, the UI shows the lock status. Also, the work item query and get services return the details of the lock in the response.

A locked work item can be reassigned. Reassignment of a work item, generally, releases the lock on that work item.

Work item locking is enabled by default.

**Relocking**

Relocking allows you to renew an already existing lock. Expiry parameters such as interval and so on are renewed based on the defaults specified in the Configurator.

If the **Default Work Item Lock Expiry Method** is set to **None** in the Configurator, a relock attempt from web services is ignored and success is returned.

The Work Item Query service returns the Lock status for a work item in response to the state of the work item. For more details, refer to **Work Item Query Service**, page 120.
Work Item Lock Service - Inputs

Command type
For both locking and relocking, the Command type needs to be set to ‘Modify’, and the entity command qualifier needs to be ‘Lock’.

<Command type="Modify">
<Workitem commandqualifier="Lock" etype="Entity">

Query Directives
NONE.

External Keys
The external keys are used to locate a work item.

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKITEMREFERENCE</td>
<td>A reference ID for each work item. This is usually available as a result of the work item query service.</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>

Work Item Lock Service - Outputs

The service attempts to lock the work item. If any errors occur (example – a non admin trying to lock someone else’s work item), they are returned as part of the response.

ResultList
For more information, refer to ResultList, on page 7.

Transaction Result
For more information, refer to TransactionResult, on page 7.
Work Item UnLock Service

This web service allows you to release the lock on a work item. The work item is unlocked and other users of the shared work item are given access to it.

A work item lock is released:

- When a work item expires or times out, it is automatically unlocked. This can be controlled by the **Enable Release of Work Item Lock on Timeout** property in the Configurator.

- When you close a work item, it is automatically unlocked and closed. Closing a work item is allowed only if there is no lock or if the lock is placed by the user who issued the close request. After the work item is closed, the lock is released. The service does not allow a locked work item to be closed, except by the owner of the lock or any user with the relevant permissions. If this happens, the following response is returned to the caller of the service:

  The work item is locked and cannot be closed.

- When a locked work item is reassigned, even if the user doing the reassignment does not own the work item.

Work Item UnLock Service - Inputs

**Command type**

The Command type needs to be set to ‘Modify’, and the entity command qualifier needs to be ‘UnLock’.

```
<Command type="Modify">
<Workitem commandqualifier="UnLock" etype="Entity">
```

**Query Directives**

NONE.
External Keys

The external keys are used to locate a work item.

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKITEMREFERENCE</td>
<td>A reference ID for each work item. This is usually available as a result of the work item query service.</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>

Work Item UnLock Service - Outputs

The service attempts to close the work item. If any errors occur (example – a non admin trying to close someone else’s work item), they are returned as part of the response.

ResultList

For more information, refer to ResultList, on page 7.

Transaction Result

For more information, refer to TransactionResult, on page 7.
Get Work Item Service

This web service allows you to get details of a work item. The details include the following:

- Work item data
- Work item lock details
- Comment history
- Primary and associated records. For each record, operational data is also included.
- Miscellaneous data from the WORKITEMDETAIL table

You can execute this web service only if you own the work item or you have the permissions to view, reassign, and close work items assigned to other users.

Get Work Item Service - Inputs

Command Type

Command type needs to be set to ‘Get’. There is no command qualifier.

<Command type="Get">
<Workitem etype="Entity">

You can specify more than one entity. To retrieve multiple work items, specify multiple transactions or multiple entities in each transaction.

Query Directives

NONE.

External Keys

Each Get request must specify exactly one external key as follows:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKITEMREFERENCE</td>
<td>Specifies the work item to be retrieved.</td>
<td>Mandatory.</td>
</tr>
</tbody>
</table>
Get Work Item Service - Outputs

Details of the specified work item are returned. The following data is returned:

- Entity data
- Comment history
- Records in work item
- Operational Data
- Context Data
- Work Item Locking

PREVIOUSREJECTIONS

Specifies whether the rejection alerts generated in the previous step (rejection) are to be retrieved.

By default, the rejection alerts generated in the previous step are retrieved.

The previous alerts are obtained based on reference step ID specified as input to the CreateWorkitem activity:

1. If reference step ID is not specified, the current step is always used irrespective of the previous rejections indicator.

2. If reference step ID is specified, and previous rejections is specified as true or not specified (default is true), reference step ID is used to get rejections.

3. If reference step ID is specified, and previous rejections is specified as false, current step id is used to get rejections.

Optional.
• Result List
• Transaction Result

Entity Data
For a list of attributes returned in the result, refer to Entity Data, on page 123.

Comment History
For more information, refer to Comment History, on page 8.

Records in Work Item
A list of primary and associated records is provided.

The primary record is identified by the \texttt{<Primary>true</Primary>} element. Associated records are those records for which a product log exists for the process log of the work item step.

The following information for each record is included:

• MASTERCATALOGNAME
• PRODUCTID
• PRODUCTIDEXT
• MODVERSION

Records are returned only if there is a primary record associated with the work item. A work item generated for batch record processing, import, data source import, mass update, or a work item which does not have any associated records will not have records as output.

The record data itself is NOT returned.

Operational Data
The operational data associated with a record includes the details of the action taken on the record and any alerts associated with the records. Possible alerts are rejections, warnings, and errors. The data is stored in the PRODUCTLOG table.

Operational data includes the following:

• Primary — Identifies the primary record in the list. Records are returned only if there is a primary record associated with the work item.

• Operation — Identifies the operation being performed. For example, EDIT PRODUCT.
• **OperationDescription** — Detailed description of the operation being performed.

• **Suboperation** — Identifies the sub-operation being performed. For example, INITIAL VERSION.

• **SuboperationDescription** — Detailed description of the sub-operation being performed.

• **OperationDate** — Date and time on which the operation was performed.

• **Alerts** — Count of the different alerts generated.
  
  — **RejectionCount**: Count of rejection entries added by an editor or approver as a part of the "Rejection" process.
  
  — **ErrorCount**: Count of errors generated when a validation rule is run. The errors are pre-computed and stored in the attributeLog table.
  
  — **WarningCount**: Count of warnings generated when a validation rule is run. The warnings are pre-computed and stored in the attributeLog table.

For example:

```xml
<OperationData>
  <Primary>true</Primary>
  <Operation>EDIT PRODUCT</Operation>
  <OperationDescription lang="en">Add/ModifyRecord</OperationDescription>
  <Suboperation>INITIAL VERSION</Suboperation>
  <SuboperationDescription lang="en">Initial Version</SuboperationDescription>
  <OperationDate>2009-02-19 09:19:13-08:00</OperationDate>
  <Message></Message>
  <Alerts>
    <RejectionCount>0</RejectionCount>
    <ErrorCount>0</ErrorCount>
    <WarningCount>0</WarningCount>
  </Alerts>
</OperationData>
```

**Context Data**

Data captured in WORKITEMDETAIL table is returned as name value pair here. Data returned depends on the work item and configuration. All data is returned as a String and all names are in upper case.

Context data includes the following attributes. However, as a workflow can be customized to capture additional data, this list is not exhaustive. Not all work items will have all the attributes listed. Only those attributes which are relevant are returned.

• **CATALOGEDITIONID** — Synchronization profile ID associated
- **CATALOGEDITIONVERSION** — synchronization profile version associated
- **Custom_*** — Custom information passed in to work item activity
- **EXPIRYTYPE** — Work item expiry type
- **INTENT** — Work item intent configured in the workflow
- **MASTERCATALOGID** — ID of the repository associated
- **MASTERCATALOGVERSION** — Version of the repository associated
- **MESSAGEID** — Any message ID is associated with work item. This is, usually, the case for work items which are created using the WaitForResponse activity.
- **NUMBEROFTIMEOUTS** — Number of timeouts for work item
- **RECORD_COUNT** — Number of records associated with the work item when the work item was created
- **RESULTDECISION** — Decision taken on the work item
- **SUCCESS_COUNT** — Number of records considered as success. The count is calculated when the work item is created.
- **SEVERITY** — Any severity associated
- **WORKITEMSTEP** — Step number
- **VALIDATIONWARNINGS** — Number of records which had warnings, only applicable for batch processing activities related work items (Import approval). The count is calculated when a work item is created.

For example:

```xml
<ContextData>
  <Attribute name="NUMBEROFTIMEOUTS" type="string">1</Attribute>
  <Attribute name="COMPAREKEYWORD" type="string">PREVIOUS_CONFIRMED_VERSION</Attribute>
  <Attribute name="EXPIRYTYPE" type="string">RELATIVE</Attribute>
  <Attribute name="INTENT" type="string">Edit</Attribute>
  <Attribute name="RECORD_COUNT" type="string">1</Attribute>
</ContextData>
```

**Work Item Locking**

For more information, refer to Work Item Lock Service, on page 130.

**ResultList**

For more information, refer to ResultList, on page 7.
Transaction Result

For more information, refer to TransactionResult, on page 7.
Work Item Reassign Service

This web service is used to reassign a work item to another user.

You can reassign a work item only if you own the work item or you have work item permissions.

When a work item is reassigned from the user interface, you can reassign the work item to specific users as controlled by the role delegation. However, when using this web service, you can reassign the work item to any valid user. The target user is not limited by role delegation setup.

Once reassigned, the work item is closed and a new work item is created for the target user. The status of the work item is the same as the original work item. For example, if the original work item was timed out, the new work item will also be timed out. If the target user has delegated the work item to other users by setting up role delegation, more than one new work items may also be created. In this case, the work items are created for final users in the delegation chain.

Any existing lock is released when a work item is reassigned only if the lock is held by the owner of the work item.

Data for all newly created work items is returned in the response XML.

Work Item Reassign Service - Inputs

Command Type

Command type needs to be set to ‘Reassign’. There is no command qualifier.

<Command type="Reassign">
<Workitem etype="Entity">

You cannot specify multiple entities in one transaction. To reassign more than one work item, create multiple transactions.

Query Directives

NONE.
External Keys

Each web service request must have external keys. The external keys have the following semantics:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKITEMREFERENCE</td>
<td>Specifies the work item to be retrieved.</td>
<td>Mandatory.</td>
</tr>
<tr>
<td>USER</td>
<td>Specifies the user to whom the work item is to be reassigned.</td>
<td>Mandatory.</td>
</tr>
</tbody>
</table>

Work Item Reassign Service - Outputs

Based on the user and WorkitemID specified in the external keys, data for all newly created work items is returned in the response XML.

Entity Data

For a list of attributes returned in the result, refer to Entity Data, on page 123.

Records in Work Item

With each work item, the list of records associated with it are also listed.

ResultList

For more information, refer to ResultList, on page 7.

Transaction Result

For more information, refer to TransactionResult, on page 7.
Get Custom Work Item Summary Service

This web service allows you to get details of work item summary for a single or multiple summary preferences. The details include the following:

- Summary Preference Name
- Level details included in the specified summary preference, such as Level name, depth, count, and its value count
- True or false value to include all users.

Only the Work Supervisor role has right to specify value for <ns:IncludeAllUsers/></ns:IncludeAllUsers> tag.

To use this web service:

1. Specify the user details such as user name, password, and enterprise name for the user.
2. Specify the summary preference name. For example, <ns:PreferenceName>Pref-1</ns:PreferenceName>.
3. Specify the level name. For example,
   
   <ns:LevelOrder>
     <ns:Name>user</ns:Name>
   </ns:LevelOrder>

   You can specify multiple levels. Use the following list of level names:
   
   - State
   - Intent
   - Repository
   - ActivityName
   - TaskType
   - Result
   - DocType
   - DocSubType
   - Actionable (Workitem Type)
   - User
   - ClosedBy
For more details on these levels, refer to the *TIBCO MDM User’s Guide.*

4. Specify true or false to indicate if the preference must sum all users or not. For example, `<ns:IncludeAllUsers>true</ns:IncludeAllUsers>`

5. Repeat from step 2 to step 4 to specify multiple summary preferences. For example, Pref-2, Pref-3, and so on.

6. Invoke the web service by specifying the endpoint URL as follows:
   
   `http://<servername>:<port>/eml/services/DataService/getWorkitemSummaryAction`

   The custom work item summary details are returned in response.

   A sample XML for a single custom work item summary is available in

   `$MQ_HOME/schema/DataService/2.0/samples/workitem/WorkitemSummary/GetCustomWorkitemSummary-request.xml`

   `$MQ_HOME/schema/DataService/2.0/samples/workitem/WorkitemSummary/GetCustomWorkitemSummary-response.xml`
A sample XML for multiple custom work item summary is available in
$MQ_HOME/schema/DataService/2.0/samples/workitem/WorkitemSummary/
GetMultipleCustomWorkitemSummary-request.xml
$MQ_HOME/schema/DataService/2.0/samples/workitem/WorkitemSummary/
GetMultipleCustomWorkitemSummary-response.xml

Get Custom Work Item Summary Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>The name of the currently logged-in user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the currently logged-in user.</td>
</tr>
<tr>
<td>Enterprise Name</td>
<td>Name of the enterprise for which the user is created. This is not required</td>
</tr>
<tr>
<td></td>
<td>if the new user is created in the same directory as the logged-in user.</td>
</tr>
<tr>
<td>Preference Name</td>
<td>Name of the work item summary preference.</td>
</tr>
<tr>
<td>Level Name</td>
<td>Name of the preference level.</td>
</tr>
<tr>
<td>IncludeAllUsers</td>
<td>Specify true or false if the work item summary is required for all users</td>
</tr>
<tr>
<td></td>
<td>or not.</td>
</tr>
</tbody>
</table>

Get Custom Work Item Summary Service - Outputs

Details of the specified custom work item summary are returned such as preference name, level depth and count, total number of work items available with the specified criteria, and then level details for each custom work item summary.

Example 14  Custom Work Item Summary with a single preference and level

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                   xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsd1/2.0">
  <soapenv:Body>
    <ns:WorkitemSummaryResponse>
      <ns:WorkitemSummaryPreference type="CUSTOM">
        <ns:Name>Pref-1</ns:Name>
        <ns:DisplayName>Pref-1</ns:DisplayName>
        <ns:LevelDepth>1</ns:LevelDepth>
        <ns:Level count="5">
          <ns:Order>0</ns:Order>
        </ns:Level>
      </ns:WorkitemSummaryPreference>
    </ns:WorkitemSummaryResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
<ns:Type>REPOSITORY</ns:Type>
<ns:DisplayName>Repository</ns:DisplayName>
<ns:Values>
  <ns:Value count="1">
    <ns:Name>2147484058</ns:Name>
    <ns:DisplayName>ACCOUNT</ns:DisplayName>
  </ns:Value>
</ns:Values>
<ns:Values>
  <ns:Value count="1">
    <ns:Name>2147484072</ns:Name>
    <ns:DisplayName>EMAIL</ns:DisplayName>
  </ns:Value>
</ns:Values>
<ns:Values>
  <ns:Value count="2">
    <ns:Name>2147484043</ns:Name>
    <ns:DisplayName>MC1</ns:DisplayName>
  </ns:Value>
</ns:Values>
<ns:Values>
  <ns:Value count="1">
    <ns:Name>2147484050</ns:Name>
    <ns:DisplayName>PERSON</ns:DisplayName>
  </ns:Value>
</ns:Values>
</ns:Level>
<ns:IncludeAllUsers>true</ns:IncludeAllUsers>
</ns:WorkitemSummaryPreference>
</ns:WorkitemSummaryResponse>
</soapenv:Body>
</soapenv:Envelope>
Example 15  Custom Work Item Summary with multiple preferences and levels

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns:WorkitemSummaryResponse
      xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
      <ns:WorkitemSummaryPreference type="CUSTOM">
        <ns:Name>Pref-1</ns:Name>
        <ns:DisplayName>Pref-1</ns:DisplayName>
        <ns:LevelDepth>2</ns:LevelDepth>
        <ns:Level count="5">
          <ns:Order>0</ns:Order>
          <ns:Type>STATE</ns:Type>
          <ns:DisplayName>State</ns:DisplayName>
          <ns:Values>
            <ns:Value count="5">
              <ns:Name>OPEN_TIMEDOUT</ns:Name>
              <ns:DisplayName>Timed out</ns:DisplayName>
            </ns:Value>
            <ns:Value count="5">
              <ns:Name>Edit</ns:Name>
              <ns:DisplayName>Edit</ns:DisplayName>
            </ns:Value>
          </ns:Values>
        </ns:Level>
        <ns:IncludeAllUsers>true</ns:IncludeAllUsers>
      </ns:WorkitemSummaryPreference>
      <ns:WorkitemSummaryPreference type="CUSTOM">
        <ns:Name>Pref-4</ns:Name>
        <ns:DisplayName>Pref-4</ns:DisplayName>
        <ns:LevelDepth>1</ns:LevelDepth>
        <ns:Level count="3">
          <ns:Order>0</ns:Order>
          <ns:Type>USER</ns:Type>
        </ns:Level>
      </ns:WorkitemSummaryPreference>
    </ns:WorkitemSummaryResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
<ns:DisplayName>User</ns:DisplayName>
<ns:Values>
  <ns:Value count="3">
    <ns:Name>2147484049</ns:Name>
    <ns:DisplayName>Smith, John</ns:DisplayName>
  </ns:Value>
</ns:Values>
</ns:Level>
<ns:IncludeAllUsers>false</ns:IncludeAllUsers>
</ns:WorkitemSummaryPreference>
</ns:WorkitemSummaryResponse>
</soapenv:Body>
</soapenv:Envelope>
Get Predefined Work Item Summary Service

This web service allows you to get details of summaries for all preferences, which are created through UI. For more details, refer to the *TIBCO MDM User’s Guide*.

To use this web service:

1. Specify the user details such as user name, enterprise name, and password for the user.

   Do not specify anything in the `<ns:PredefinedSummary></ns:PredefinedSummary>` tag, else an error message is returned in response.

2. Invoke the web service by specifying the endpoint URL as follows:
   
   ```
   http://<servername>:<port>/eml/services/DataService/getWorkitemSummaryAction
   ```

   The work item summary details for all predefined summary preferences are returned in response.

A sample XML is available in

- `$MQ_HOME/schema/DataService/2.0/samples/workitem/WorkitemSummary/GetPredefinedWorkitemSummary-request.xml`
- `$MQ_HOME/schema/DataService/2.0/samples/workitem/WorkitemSummary/GetPredefinedWorkitemSummary-response.xml`

Get Predefined Work Item Summary Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>The name of the currently logged-in user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the currently logged-in user.</td>
</tr>
<tr>
<td>Enterprise Name</td>
<td>Name of the enterprise for which the user is created. This is not required if the new user is created in the same directory as the logged-in user.</td>
</tr>
<tr>
<td>Predefined Summary</td>
<td>Refers to all defined work item summary preferences.</td>
</tr>
</tbody>
</table>
Get Predefined Work Item Summary Service - Outputs

Details of the specified work item summary preference are returned, such as repository name or user name and its level details. For example,

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
  <soapenv:Body>
    <ns:WorkitemSummaryResponse
      xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
      <ns:WorkitemSummaryPreference type="PREDEFINED">
        <ns:Name>Pref-1</ns:Name>
        <ns:DisplayName>Pref-1</ns:DisplayName>
        <ns:LevelDepth>1</ns:LevelDepth>
        <ns:Level count="3">
          <ns:Order>0</ns:Order>
          <ns:Type>REPOSITORY</ns:Type>
          <ns:DisplayName>Repository</ns:DisplayName>
          <ns:Values>
            <ns:Value count="1">
              <ns:Name>2147484058</ns:Name>
              <ns:DisplayName>ACCOUNT</ns:DisplayName>
            </ns:Value>
          </ns:Values>
          <ns:Values>
            <ns:Value count="1">
              <ns:Name>2147484072</ns:Name>
              <ns:DisplayName>EMAIL</ns:DisplayName>
            </ns:Value>
          </ns:Values>
          <ns:Values>
            <ns:Value count="1">
              <ns:Name>2147484050</ns:Name>
              <ns:DisplayName>PERSON</ns:DisplayName>
            </ns:Value>
          </ns:Values>
        </ns:Level>
        <ns:IncludeAllUsers>false</ns:IncludeAllUsers>
      </ns:WorkitemSummaryPreference>
    </ns:WorkitemSummaryResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
Chapter 4  Administration Services

This chapter describes the usage and customization of administration services.

Topics

- Login Service, page 152
- Logout Service, page 153
- Get Entitlement Service, page 154
- Create Enterprise Service, page 157
- Get Enterprise List Service, page 159
- Create User Service, page 161
- Modify User Service, page 166
- Change Password Service, page 172
- Delete User Service, page 174
- Get User List Service, page 176
- Get Role List Service, page 178
Login Service

This service allows you to login to TIBCO MDM after authenticating the specified credentials. This service also creates a session and returns the sessionId. The sessionId can be used by any client such as TIBCO BusinessWorks or TIBCO General Interface. The session is maintained by the URL rewriting mechanism. Also, if cookies are enabled for a browser-based soap client, it is not mandatory to send the sessionId in URL.

To use this web service:

1. Specify the user name, password, and enterprise for the user.
2. Invoke the web service by specifying the endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/AdminService/loginAction
   
   A new HTTP session is created.

The specified user must be able to log on to TIBCO MDM successfully and the response returns the session ID.

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/admin/Login-Request.xml

$MQ_HOME/schema/DataService/2.0/samples/admin/Login-Response.xml

Many web services in TIBCO MDM are SSO enabled. For more details, refer to, Authentication, on page 18.

Login Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
</tbody>
</table>

Login Service - Outputs

A success or failure message is returned. For more information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.
Logout Service

This service allows you to logout from the user session for the maintained session. The session is maintained either by URL re-writing mechanism or through cookies.

To use this web service:

1. Specify the user name, password, and enterprise for the user.
2. Invoke the web service.

   — For a cookies-enabled browser client, specify the endpoint URL as:
     
     ```plaintext
     http://<servername>:<port>/eml/services/AdminService/logout
     Action
     ```

   — For a client where cookies are disabled or for a client which does not operate with a browser (for example: TIBCO BusinessWorks), use jsessionid as a part of the URL:
     
     ```plaintext
     http://<servername>:<port>/eml/services/AdminService/logoutA
     ction;jsessionid=T2yJKLDPzSY4CjcbzpG3gGm2qlqmGb4Mz1GpSsv5Gsw
     xnPZXKBKc!-1102618730!1242055503359
     ```

   The specified user logs out from TIBCO MDM.

   A sample XML is available in

   $MQ_HOME/schema/DataService/2.0/samples/admin/Logout-Request.xml
   $MQ_HOME/schema/DataService/2.0/samples/admin/Logout-Response.xml

Logout Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
</tbody>
</table>

Logout Service - Outputs

If a session is not present, an "Invalid session" message is returned. For more information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.
Get Entitlement Service

This service allows you to retrieve entitlement information for a specified user. Entitlement information includes user permissions, roles assigned, and functions assigned to the user.

This service lists permissions at the user-level only and not at a role-level. If a permission is denied at a role level and you run the Get Entitlement service, the permission is displayed as allowed even though it is denied at the role-level.

You do not require any special privileges to retrieve your own entitlement information. For information on authentication of Get Entitlement Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify the user name, password, and enterprise for the user.
2. Invoke the web service by specifying the endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/AdminService/getEntitlementAction

A sample XML is available in
   
   $MQ_HOME/schema/DataService/2.0/samples/admin/GetEntitlement-request.xml
   
   $MQ_HOME/schema/DataService/2.0/samples/admin/GetEntitlement-response.xml

Restricting Access to Entitlement Information

To restrict an user from getting entitlement information, run the following query in the SQL editor:

1. Create the function in the function table:
   
   ```sql
   insert into FUNCTION values ((select max(id)+1 from FUNCTION), 'allow_view_entitlement', 'Allow viewing entitlement information', ' allow_view_entitlement');
   ```

2. Identify for which role the function is to be restricted:
   
   ```sql
   insert into ROLE2FUNCMAP(ID, ROLEID, FUNCID, NAME, PRIORITY) values((select max(id)+1 from ROLE2FUNCMAP), ROLEID, FUNCID, 'allow_view_entitlement', PRIORITY);
   ```

3. To inactivate or disable the function allow_view_entitlement for a particular user, restart the server after running the query. Similarly, when you want to enable the allow_view_entitlement function for a particular user, restart the server after running the query.
Get Entitlement Service - Inputs

The following inputs are required when entitlement information for other users needs to be retrieved. If this input is not specified, entitlement information for the currently logged in user is displayed.

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>User Name for Entitlement</td>
<td>Name of the user for whom entitlement information is to be retrieved.</td>
</tr>
<tr>
<td>Enterprise for Entitlement</td>
<td>Name of the enterprise.</td>
</tr>
</tbody>
</table>

Get Entitlement Service - Outputs

The response message provides information about:

- **All roles assigned** — For example, the following information is returned for each role.

  `<ns:Roles>
   <ns:Role id="34018">
     <ns:Name>Admin</ns:Name>
     <ns:Description lang="en">Default Admin Role</ns:Description>
   </ns:Role>
  </ns:Roles>

- **All accessible functions** — For example, the following information is returned for each function.

  `<ns:Functions>
   <ns:Function id="61">
     <ns:Name>adminmenu_roles</ns:Name>
     <ns:Description lang="en">Roles menu</ns:Description>
   </ns:Function>
  </ns:Functions>`
• **Access permissions for all resources** — For example, the following information is returned for permissions. Note that the service calculates cumulative user permission for a resource. For example, user "A" is assigned user level permission to allow browse for resource repository "MC1". User "A" is also assigned two roles, "Catalog Manager" and "Catalog Edition" and both roles are denied permission for resource "MC1". Only the permissions at the user level are populated in the response file. Also, note that, if "Full control" permission is assigned to a given resource, the rest of the permissions are not populated in the response.

```xml
<ns:Resource id="34190" resourceType="Repository">
  <ns:Name>MC1</ns:Name>
  <ns:Accesses>
    <ns:Access id="A">
      <ns:Description lang="en">Full Control</ns:Description>
      <ns:Allowed>Y</ns:Allowed>
    </ns:Access>
  </ns:Accesses>
</ns:Resource>
```

For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.
Create Enterprise Service

This service allows you to create an enterprise in TIBCO MDM.

For information on authentication of Create Enterprise Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify the user details such as user name, password, and enterprise for the user.
2. Specify the enterprise details such as enterprise name, enterprise internal name, and its industry vertical.
3. Invoke the web service by specifying the endpoint URL as follows:
   http://<servername>:<port>/eml/services/AdminService/adminAction/createEnterpriseAction

   The specified enterprise is created in TIBCO MDM.

A sample XML is available in
   $MQ_HOME/schema/DataService/2.0/samples/admin/CreateEnterprise_Request.xml
   $MQ_HOME/schema/DataService/2.0/samples/admin/CreateEnterprise_Response.xml

Create Enterprise Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise of the Super User, Work Supervisor, or Administrator.</td>
</tr>
<tr>
<td>Name (Mandatory)</td>
<td>Name of the enterprise that is to be created.</td>
</tr>
<tr>
<td>Internal Name (Mandatory)</td>
<td>Internal name of the enterprise.</td>
</tr>
</tbody>
</table>
### Create Enterprise Service - Outputs

If an enterprise is created, the "Enterprise Created Successfully" message is returned.

For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.
Get Enterprise List Service

This service allows you to get a list of enterprises that are created in TIBCO MDM. When deploying rulebases and workflows from TIBCO MDM Studio, administrator needs to specify a user list of enterprises to which the rulebase and workflow can be deployed. This service is used in the Network Deployment wizard.

For information on authentication of Get Enterprise Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify the user details such as user name, enterprise name, and password for the user.

2. Invoke the web service by specifying the endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/AdminService/GetEnterpriseListAction

   A list of all enterprises is returned in response. However, if the administrator or work supervisor executes this web service, a list of a particular enterprise that he/she belongs to is returned.

   If the user does not have access to the Administration menu, an error message is returned in response.

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/admin/GetEnterpriseList-Request.xml
$MQ_HOME/schema/DataService/2.0/samples/admin/GetEnterpriseList-Response.xml

Get Enterprise List Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>The name of the currently logged-in user.</td>
</tr>
<tr>
<td>Enterprise Name</td>
<td>Name of the enterprise for which the user is created.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the currently logged-in user.</td>
</tr>
</tbody>
</table>
Get Enterprise List Service - Outputs

A list of enterprises is returned. For each enterprise, the enterprise name and industry vertical details are returned.

If other than Administrator and Super User invokes this web service, the authentication failure message is returned. For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.
Create User Service

Using this web service, you can create a user in an existing Enterprise.

For information on authentication of Create User Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify user details such as, the user name, password, and enterprise for the user.

2. Specify new user details that you want to create such as, the user name, enterprise, security type, first name, last name, middle name, and password.


4. Specify role details of a user such as, the repository editor, repository manager, and so on.

5. Specify local information such as, the language, country, date format, time format, timestamp format, and timezone.

6. Invoke the web service by specifying the endpoint URL as follows:

   http://<servername>:<port>/eml/services/AdminService/userAdminAction/createUserAction

   The new user is created in an existing enterprise.

The Username can accept the following special characters as a part of the name:

* (asterisk)
. (period)
- (minus)
@ (at)
/ (Forward Slash)
\ (Backward Slash)

These special characters are accepted when you enter the user name while adding permissions, roles, and users. These can be used when referring to the user name in a dataset of a File Watcher and also while referring to the user name in a workflow.
The following are sample XML files:
$MQ_HOME/schema/DataService/2.0/samples/admin/CreateUser_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/admin/CreateUser_Response.xml
$MQ_HOME/schema/DataService/2.0/samples/admin/CreateUser_encryptedPassword-Request.xml
$MQ_HOME/schema/DataService/2.0/samples/admin/CreateUser_encryptedPassword-Response.xml

Create User Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserName (Mandatory)</td>
<td>Login ID of the Super User, Work Supervisor, or Administrator.</td>
</tr>
<tr>
<td>Password (Mandatory)</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise (Mandatory)</td>
<td>Name of the enterprise of the Super User, Work Supervisor, or Administrator.</td>
</tr>
<tr>
<td>UserName (Mandatory)</td>
<td>Unique name for the new user.</td>
</tr>
<tr>
<td>Firstname (Mandatory)</td>
<td>First name of the user.</td>
</tr>
<tr>
<td>Lastname (Mandatory)</td>
<td>Last name of the user.</td>
</tr>
<tr>
<td>Password (Mandatory)</td>
<td>Password for the new user name.</td>
</tr>
<tr>
<td>Enterprise (Mandatory)</td>
<td>Name of the enterprise for which the user is created.</td>
</tr>
</tbody>
</table>

**Note:** This input is not mandatory, if you select the Security Type as LDAP.

**Note:** This input is not required if you want to create a new user in the same enterprise where the user is logged-in.
<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsEncrypted</td>
<td>Specify <strong>true</strong> if you want the password in the encrypted format.</td>
</tr>
<tr>
<td></td>
<td>Specify <strong>false</strong> if you do not want the password in the encrypted format.</td>
</tr>
<tr>
<td>Password</td>
<td>Specify the password that you want to update.</td>
</tr>
<tr>
<td>(Mandatory)</td>
<td><strong>Note</strong>: If you have specified <strong>true</strong> for the IsEncrypted attribute, specify the encrypted password. While logging into TIBCO MDM from UI as well as from the web service, specify the decrypted form of a password. If you have specified <strong>false</strong> for the IsEncrypted attribute, specify the plain text password. While logging into TIBCO MDM from UI as well as from the web service, specify the as is password mentioned in the request file.</td>
</tr>
<tr>
<td>Securitytype</td>
<td>Type of security.</td>
</tr>
<tr>
<td>(Mandatory)</td>
<td>Allowed Values are <strong>Password</strong> or <strong>LDAP</strong>. If you select the LDAP security type, the user must already exist in the LDAP directory server.</td>
</tr>
<tr>
<td>Roles</td>
<td>Role for the user name should be specified from the existing roles specified for the Enterprise.</td>
</tr>
<tr>
<td>(Mandatory)</td>
<td></td>
</tr>
<tr>
<td>Language</td>
<td>Only a language supported by TIBCO MDM is accepted. This input is mandatory if country is specified. Country can be kept blank with language value specified.</td>
</tr>
<tr>
<td>(Optional)</td>
<td></td>
</tr>
<tr>
<td>Country</td>
<td>Name of the country of the new user. Language is mandatory if country is specified.</td>
</tr>
<tr>
<td>(Optional)</td>
<td></td>
</tr>
</tbody>
</table>
### Input Description

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DateFormat</strong></td>
<td>Only a date format value supported by TIBCO MDM is accepted. By default, <strong>DD-MM-YY</strong> value is specified in the request XML file. You can change the specified date format. The supported date format values are:</td>
</tr>
<tr>
<td>(Optional)</td>
<td>• <strong>MM/DD/YYYY</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>DD-MON-YYYY</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>MM/DD/YY</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>DDMMYYYY</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>YYYY-MM-DD</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>YYYY/MM/DD</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>DD-MM-YYYY</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>DD/MM/YYYY</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>DD-MM-YY</strong></td>
</tr>
<tr>
<td><strong>TimeFormat</strong></td>
<td>Only a time format value supported by TIBCO MDM is accepted. By default, <strong>hh:mm:ss</strong> value is specified in the request XML file. You can change the specified time format. The supported time format values are:</td>
</tr>
<tr>
<td>(Optional)</td>
<td>• <strong>hh:mm:ss</strong> (24 hours)</td>
</tr>
<tr>
<td></td>
<td>• <strong>hh:mm:ss AM/PM</strong> (12 hours)</td>
</tr>
</tbody>
</table>
Create User Service - Outputs

If a user is created, the "User Created Successfully" message is returned.

For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimestampFormat</td>
<td>Only a timestamp format value supported by TIBCO MDM is accepted. By default, YYYY-MM-DD hh:mm:ss.S value is specified in the request XML file. You can change the specified timestamp format. The supported timestamp format values are:</td>
</tr>
<tr>
<td>(Optional)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• YYYY-MM-DD hh:mm:ss.S - Time is displayed in 24 hours format with milliseconds. .S indicates milliseconds.</td>
</tr>
<tr>
<td></td>
<td>• YYYY-MM-DD hh:mm:ss.S AM/PM - Time is displayed in 12 hours format with milliseconds.</td>
</tr>
<tr>
<td></td>
<td>• YYYY-MM-DD hh:mm:ss.S z - Time is displayed in 24 hours format with milliseconds and time zone. Z indicates time zone.</td>
</tr>
<tr>
<td></td>
<td>• YYYY-MM-DD hh:mm:ss.S AM/PM z - Time is displayed in 12 hours format with milliseconds and time zone.</td>
</tr>
<tr>
<td>Timezone</td>
<td>Only time zone ID is accepted. For example, Asia/Kolkata. By default, Asia/Tokyo value is specified in the request XML file. You need to specify the time zone ID from the 151 time zones supported list. The supported time zones are listed in a text file. The UserProfileTimeZoneList.txt and AllTimeZoneList.txt files. The files are located at $MQ_HOME\mdm\8.3\common\standard\samples\timezone folder.</td>
</tr>
</tbody>
</table>
Modify User Service

Using this web service, you can modify the information of an existing user. As an administrator, you can modify the following user information:

- User credentials, such as, the first name, last name, middle name, and password
- The locale information, date format, time format, timestamp format, time zone and partitioning key
- The roles assigned to the user

Only Administrator and Work Supervisor can modify the roles.

- Delegation profile including users
  
  All fields are validated similar to the Create User web service. Only the users’ list specified in the delegation profile is not validated against the delegation setup for roles.

For information on authentication of Modify User Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify user details such as, user name, password, and an enterprise for the user.
2. Modify user details such as, locale settings, and role delegation information.
3. Invoke the web service by specifying the endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/AdminService/userAdminAction/modifyUserAction

   The user is modified in an enterprise.

The following are sample XML files:

$MQ_HOME/schema/DataService/2.0/samples/admin/ModifyUser_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/admin/ModifyUser_Response.xml
Modify User Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ModifyUserDetails</td>
<td></td>
</tr>
<tr>
<td>UserName (Mandatory)</td>
<td>The login ID of the Super User, Work Supervisor, or Administrator. Users can also specify their Login ID. However, they can only modify their user name and locale information.</td>
</tr>
<tr>
<td>Enterprise (Mandatory)</td>
<td>The name of the enterprise of the Super User, Work Supervisor, or Administrator. Or The name of the enterprise to which the modifying user belongs.</td>
</tr>
<tr>
<td>Firstname (Mandatory)</td>
<td>The first name of the user.</td>
</tr>
<tr>
<td>Lastname (Mandatory)</td>
<td>The last name of the user.</td>
</tr>
<tr>
<td>Middlename (Mandatory)</td>
<td>The middle name of the user.</td>
</tr>
<tr>
<td>PartitionKey</td>
<td>A parameter used only in special circumstances. The PartitionKey contains the partition number for the user in case the user based partitioning is implemented.</td>
</tr>
<tr>
<td>IsEncrypted</td>
<td>Specify true if you want the password in the encrypted format. Specify false if you do not want the password in the encrypted format.</td>
</tr>
</tbody>
</table>
## Password

(Mandatory)

Specify the password that you want to update.

**Note:** If you have specified `true` for the `IsEncrypted` attribute, specify the encrypted password. While logging into TIBCO MDM from UI as well as from the web service, specify the decrypted form of a password.

If you have specified `false` for the `IsEncrypted` attribute, specify the plain text password. While logging into TIBCO MDM from UI as well as from the web service, specify the as is password mentioned in the request file.

### Roles

<table>
<thead>
<tr>
<th>Roles</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roles</td>
<td>The role for the user name. Specify an existing role from the enterprise. If you do not specify a role, it retrieves the role that was previously assigned to the user.</td>
</tr>
</tbody>
</table>

### LocaleSettings

<table>
<thead>
<tr>
<th>Language</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Optional)</td>
<td>Only a language supported by TIBCO MDM is accepted. This input is mandatory if the country is specified. You can keep the Country value blank and specify the language value.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Country</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Optional)</td>
<td>The name of the country of the new user.</td>
</tr>
</tbody>
</table>
### DateFormat (Optional)

Only a date format value supported by TIBCO MDM is accepted. By default, **DD-MM-YY** value is specified in the request XML file. You can change the specified date format. The supported date format values are:

- **MM/DD/YYYY**
- **DD-MON-YYYY**
- **MM/DD/YY**
- **DDMMYYYY**
- **YYYY-MM-DD**
- **YYYY/MM/DD**
- **DD-MM-YYYY**
- **DD/MM/YYYY**
- **DD-MM-YY**

### TimeFormat (Optional)

Only a time format value supported by TIBCO MDM is accepted. By default, the **hh:mm:ss** value is specified in the request XML file. You can change the specified time format. The supported time format values are:

- **hh:mm:ss** (24 hours)
- **hh:mm:ss AM/PM** (12 hours)

### TimestampFormat (Optional)

Only a timestamp format value supported by TIBCO MDM is accepted. By default, the **YYYY-MM-DD hh:mm:ss.S** value is specified in the request XML file. You can change the specified timestamp format. The supported timestamp format values are:

- **YYYY-MM-DD hh:mm:ss.S** - Time is displayed in a 24 hour format with milliseconds. **.S** indicates milliseconds.
- **YYYY-MM-DD hh:mm:ss.S AM/PM** - Time is displayed in a 12 hour format with milliseconds.
- **YYYY-MM-DD hh:mm:ss.S z** - Time is displayed in a 24 hour format with milliseconds and time zone. **Z** indicates time zone.
- **YYYY-MM-DD hh:mm:ss.S AM/PM z** - Time is displayed in a 12 hour format with milliseconds and time zone.
### Timezone

(Optional)

Only the time zone ID is accepted. For example, Asia/Kolkata. By default, the Asia/Tokyo value is specified in the request XML file.

Specify the time zone ID from the supported time zones listed in the following files:

- `$MQ_HOME/mdm/8.3/common/standard/samples/timezone/UserProfileTimeZoneList.txt`
- `$MQ_HOME/mdm/8.3/common/standard/samples/timezone/AllTimeZoneList.txt`

### Delegation Profile

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DelegationActive</td>
<td>Specify <strong>true</strong> to activate delegation to the specified users. After the delegation is activated, all work items are automatically delegated to the specified users. Specify <strong>false</strong> to deactivate delegation of the specified users.</td>
</tr>
<tr>
<td>StartDate</td>
<td>Specify the start date when the delegation is to be activated.</td>
</tr>
<tr>
<td>EndDate</td>
<td>Specify the end date when the delegation is to be deactivated.</td>
</tr>
<tr>
<td>Notify</td>
<td>Specify <strong>true</strong> to notify the users about the work items delegated to them. After the work item is delegated to another user, an alert message is displayed when the user logs in to the company.</td>
</tr>
<tr>
<td>AutoRevertFlag</td>
<td>Specify <strong>true</strong> or <strong>false</strong>. If you specify <strong>true</strong>, the delegation remains in effect even after the delegation end date has passed.</td>
</tr>
<tr>
<td>Users</td>
<td>Specify name of the users to whom you want to delegate the work items based on their role.</td>
</tr>
</tbody>
</table>

**UserDefinedAttributes**
Modify User Service - Outputs

If you modify a user, the `User information modified` message is returned.

For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttributeHelp</td>
<td>Specify <code>Y</code> to display the repository attribute Help. To edit records, move the cursor over an attribute; this displays a Help icon. Specify <code>N</code> if you do not want to display the repository attribute Help.</td>
</tr>
</tbody>
</table>
Change Password Service

Using this web service, you can change the password of an user without providing any other user information for modification. If you are an Administrator or a Work Supervisor, you can modify your own password and also the password of other users.

For information on authentication of Modify User Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify user details such as user name, password, and an enterprise for the user.
2. Specify the change password details such as, a value for the <IsEncrypted> and <Password> attributes.
3. Invoke the web service by specifying the endpoint URL as follows:

   http://<servername>:<port>/eml/services/AdminService/userAdminAction/changepasswordAction

   The password of a user is changed.

The following are sample XML files:

$MQ_HOME/schema/DataService/2.0/samples/admin/ChangePassword_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/admin/ChangePassword_Response.xml

Change Password Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserInfo</td>
<td></td>
</tr>
<tr>
<td>UserName (Mandatory)</td>
<td>The login ID of the Super User, Work Supervisor, or Administrator. Users can also specify their login ID. However, they can only change their own password.</td>
</tr>
<tr>
<td>Password (Mandatory)</td>
<td>The password for the specified user name.</td>
</tr>
</tbody>
</table>
### Change Password Service

#### Inputs

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise (Mandatory)</td>
<td>The name of the enterprise of the Super User, Work Supervisor, or Administrator. The name of the enterprise to which the user belongs.</td>
</tr>
</tbody>
</table>

#### ChangePasswordDetails

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsEncrypted</td>
<td>Specify <code>true</code> if you want the password in the encrypted format.</td>
</tr>
<tr>
<td></td>
<td>Specify <code>false</code> if you do not want the password in the encrypted format.</td>
</tr>
<tr>
<td>Password (Mandatory)</td>
<td>Specify the password that you want to update.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If you have specified <code>true</code> for the <code>IsEncrypted</code> attribute, specify the encrypted password. While logging into TIBCO MDM from UI as well as from the web service, specify the decrypted form of a password. If you have specified <code>false</code> for the <code>IsEncrypted</code> attribute, specify the plain text password. While logging into TIBCO MDM from UI as well as from the web service, specify the as is password mentioned in the request file.</td>
</tr>
</tbody>
</table>

#### Change Password Service - Outputs

The **User information modified** message is returned if:

- the password is changed with text modification.
- the password is changed with an encrypted password.

The **Password updated** message is returned if:

- the password contains a space.
- the `true` or `false` value is specified for the `<IsEncrypted>` attribute.

For information on failure messages, refer to *Table 11, Faults of Administrator Web Services, on page 26.*
Delete User Service

This service allows you to delete a user in an existing Enterprise.

For information on authentication of Delete User Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify the user details such as user name, password, and enterprise for the user.

2. Specify the user name and its enterprise that you want to delete.

3. Invoke the web service by specifying the endpoint URL as follows:
   
   \[
   \text{http://<servername>:<port>/eml/services/AdminService/userAdminAction/deleteUserAction}
   \]

   The specified user is deleted from an existing enterprise.

A sample XML is available in:

\[
\text{$MQ\_HOME/schema/DataService/2.0/samples/admin/DeleteUser\_Request.xml}
\]

\[
\text{$MQ\_HOME/schema/DataService/2.0/samples/admin/DeleteUser\_Response.xml}
\]

Delete User Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Login ID of the Super User, Work Supervisor, or Administrator.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise of the Super User, Work Supervisor, or Administrator.</td>
</tr>
<tr>
<td>User Name</td>
<td>Name of the user to be deleted.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The user should not have any open work item, else an error is returned.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This input is not required if you want to delete a user from the same enterprise where the user is logged-in.</td>
</tr>
</tbody>
</table>
Delete User Service - Outputs

If a user is deleted, the User Deleted Successfully message is returned.

For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.
Get User List Service

This service allows you to get a list of all defined users for an enterprise. You can also retrieve a list of users based on the <UserName> filter criteria. For example, if you want to retrieve a list of users that starts with the ‘a’ alphabet.

For information on authentication of Get User List Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify the user name, password, and enterprise for the user.
2. Specify the true or false value for the Exact Search attribute.
3. Invoke the web service by specifying the endpoint URL as follows:
   
   ```
   http://<servername>:<port>/eml/services/AdminService/GetUserListAction
   ```

   A list of all defined users for the specified enterprise is returned in response.

A sample XML is available in:

- `$MQ_HOME/schema/DataService/2.0/samples/admin/GetUserList-request.xml`
- `$MQ_HOME/schema/DataService/2.0/samples/admin/GetUserList-response.xml`

Get User List Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>The name of the currently logged-in user.</td>
</tr>
<tr>
<td>Enterprise Name</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the currently logged-in user.</td>
</tr>
</tbody>
</table>
| Exact Search (Optional) | Specify true or false value.   
  • True returns the users list exactly matching with the filter criteria.   
  • False returns the users list that contains the specified filter criteria. |
| User Name (Optional) | Name of the user that you want to retrieve.                               |
Get User List Service - Outputs

A list of all defined users is returned for the specified enterprise. For each user, the User ID, first name, middle name, last name, enterprise name, language, locale, date and time format, modification member ID, modification date, modification version, active and delegation active is returned.

For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Name</td>
<td>First name of the user that you want to retrieve. (Optional)</td>
</tr>
<tr>
<td>Last Name</td>
<td>Last name of the user that you want to retrieve. (Optional)</td>
</tr>
</tbody>
</table>
Get Role List Service

This service allows you to get a list of roles for all defined users for the specified enterprise. You can also retrieve a list of roles based on <RoleName> filter criteria. For example, you may want to get a list of Repository Manager roles.

For information on authentication of Get Role List Service, refer to Table 6, Authentication of Administrator Web Services, on page 20.

To use this web service:

1. Specify the user name, password, and enterprise for the user.
2. Specify the true or false value for the Exact Search attribute.
3. Specify the role name for filter criteria. For example, you can specify 'Repository' to retrieve a list of roles that start with 'Repository'. The Role Name is an optional parameter.
4. Invoke the web service by specifying the endpoint URL as follows:
   
   ```
   http://<servername>:<port>/eml/services/AdminService/GetRoleListAction
   ```

   A list of roles of all defined users for the specified enterprise is returned. If you have specified the Repository value for the <RoleName> attribute, a list of roles that start with Repository are returned, such as Repository Manager, Repository Editor, and Repository Approver.

   A sample XML is available in
   
   `$MQ_HOME/schema/DataService/2.0/samples/admin/GetRoleList-request.xml`
   
   `$MQ_HOME/schema/DataService/2.0/samples/admin/GetRoleList-response.xml`

Get Role List Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>The name of the currently logged-in user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the user name.</td>
</tr>
<tr>
<td>Enterprise Name</td>
<td>Name of the enterprise.</td>
</tr>
</tbody>
</table>
Get Role List Service - Outputs

A list of roles for all defined users and for the specified enterprise is returned. For each role, the Enterprise ID, its name, description, type, active, modification member ID, modification date, modification version, and organization type is returned.

For information on failure messages, refer to Table 11, Faults of Administrator Web Services, on page 26.
Chapter 5  

**Data Services**

This chapter describes the usage and customization of data services.

**Topics**

- Text Search Service, page 182
- Get DataSource List Service, page 191
- Get Repository List Service, page 193
- Initiate Synchronization Service, page 199
- Upload Metadata Service, page 201
- Extract Data Service, page 203
- GetClassificationSchemeList, page 210
- GetClassificationSchemeAttributeList, page 213
- GetClassificationCodeHierarchy, page 216
Text Search Service

The Text Search web service, available as `cimDataService.wsdl`, allows you to search for human recognizable terms in one or more TIBCO MDM repositories as you would do on an Internet Search Engine. The WSDL is available at `$MQ_HOME/schema/DataService/2.0/cimDataService.wsdl`. This WSDL can be used to generate client stubs for various client languages.

The Text Search web service indexes the text in a record and stores them as key terms in a data index. For example, if you search for records with a description "The product is in red color", the search utility looks for the key terms, "product", "red", and "color". Articles, prepositions, pronouns, and other such fillers are not considered as key terms. The Text Search index contains the Record ID and the latest version of the record.

- Text search is offered as a separate service, resulting in a separate WSDL and a different endpoint address. This is in contrast to the other services which are multiplexed in the same WSDL and use different message parameters to invoke a specified service.
- Only the latest confirmed or unconfirmed record are searchable through the text search service.
- The search expression must not exceed 1024 bytes or 10 terms (words).
- This service does not support query of deleted records.

This web service also allows you to conduct similarity searches, that is, searches which are close to an existing indexed term (for example, due to a spelling error) are found and returned. A Fuzzy search or similarity search is more expensive than standard text search and should be used in a more focused way than the 'exact' text search, by limiting the repositories or attributes.

For information on authentication of Text Search Service, refer to Table 7, Authentication of Data Web Services, on page 22.

Using this web service, you can:
- Search for text against the entire index (all indexed catalogs)
- Search for text search against the entire index (all indexed catalogs) retrieving only a limited client determined set (paged query)
- Search for text search against a single repository
- Search for text against a single repository retrieving only a limited client determined set (paged query)
- Fuzzy search against a single repository
**Enabling Text Search**

To enable text search, set the following properties in Configurator (Initial Config > Repository):

- **Text Indexing Enabled**: Go to Initial Config > Advanced > Repository, set this property to either OFFLINE or ONLINE. The default is NONE.

- **Text Indexing Receiver Pool Size**: Go to Member1 > Advanced > Async Task Management, specify the number of threads processing in parallel. Set this property to 0 to enable the indexing and to 0 to disable the automatic text indexing. Do not set the property to a value greater than 1.

Additionally, you can also specify the following properties in Configurator:

- **Default Similarity Text Search Score**: Go to Initial Config > Miscellaneous, specify the default similarity (fuzzy) score which is needed to return a matching record. The value must be between 0 and 1.0. The default value is 0.8.

**Warning**

Lowering the default fuzzy score may affect performance and result in irrelevant records.

**Text Search Service - Inputs**

This web service contains a single web service operation, textSearchAction.

```xml
<wsdl:portType name="DataServicePortType">
<wsdl:operation name="textSearchAction">
<wsdl:input message="tns:textSearchRequest"/>
<wsdl:output message="tns:textSearchResponse"/>
</wsdl:operation>
</wsdl:portType>
```
The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Name of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the user.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
</tbody>
</table>

**Text Search Service - Outputs**

A list of records matching the search criteria is provided as an output in the `<Records>` element.

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.

**Types of Text Search Services**

You can specify following types of text search web services:

- Text Search Exact Query Service
- Text Search Fuzzy Query Service
- Index Entity Fuzzy Query Service

The sample XML files for each Text Search web service are available at $MQ_HOME/schema/DataService/2.0/samples/repository/textSearch/

- TextSearch-AttributeFuzzyQuery-Request(With Weight).xml
- TextSearch-AttributeFuzzyQuery-Request(Without Weight).xml
- TextSearch-AttributeFuzzyQuery-Response(With Weight).xml
- TextSearch-AttributeFuzzyQuery-Response(Without Weight).xml
- TextSearch-ExactQuery-Request.xml
- TextSearch-ExactQuery-Response.xml
- TextSearch-FuzzyQuery-Request.xml
- TextSearch-FuzzyQuery-Response.xml
- TextSearch-IndexEntityFuzzyQuery-Request(With RA).xml (RA stands for Relationship Attributes)
- TextSearch-IndexEntityFuzzyQuery-Request.xml
- TextSearch-IndexEntityFuzzyQuery-Response(With RA).xml (RA stands for Relationship Attributes)
- TextSearch-IndexEntityFuzzyQuery-Response.xml
Text Search Exact Query Service

For Text Search Exact Query web service, in addition to the mandatory inputs, specify the following tags in the request:

- **Repository location indicator**: Indicates a specific repository search.
  - Specific repository
    ```xml
    <ns:RepositoryNames>
    <ns:RepositoryName>CUSTOMER</ns:RepositoryName>
    </ns:RepositoryNames>
    ```

- **Search string**: Indicates the exact search by specifying the string.
  - Exact Search
    ```xml
    <ns:ExactSearchExpression>FIRSTNAME</ns:ExactSearchExpression>
    ```

- **Record Count**: You can specify a number of matching records specified in a repository that you want to display in the response.
  ```xml
  <ns:ReturnCount>
  <ns:StartCount>1</ns:StartCount>
  <ns:TotalCount>1</ns:TotalCount>
  </ns:ReturnCount>
  ```
  - Start Count represents the sequential number of matching record that you want to display in the response. For example, if there are three matching records in a repository and you want to view the second matching record. You can specify `<ns:StartCount>2</ns:StartCount>`.
  - Total Count represents the total number of matching records that you want to display in the response. For example, if there are five matching records in a repository and you want to view all five records, you can specify `<ns:TotalCount>5</ns:TotalCount>`.
  - If you do not specify record count, all matching records specified in the Exact Search criteria are displayed.
  - You must specify the value for both `StartCount` and `TotalCount`. If any missing value for any one of these tag; results in failure response.

- You can search in a specific repository only.
- Do not restrict an exact text search to specific attributes.
Text Search Fuzzy Query Service

For Text Search Fuzzy Query web service, in addition to the mandatory inputs, specify the following tags in the request:

- Repository location indicator: Specify the repository name and its attributes.
  - Specifying an attribute set (used for a Similarity Search only)
    `<RepositoryNames>
    <RepositoryName attributes="FirstName">Person</RepositoryName>
    </RepositoryNames>`
  - Specific repository set (used for both Exact and Similarity Search)
    `<RepositoryNames>
    <RepositoryName>Person</RepositoryName>
    </RepositoryNames>`
  - Similarity (fuzzy) Search
    `<FuzzySearchExpression similarityScore="0.95">John</FuzzySearchExpression>`

You can search a specific repository. You can also limit the results based on an attribute set.

- You cannot perform a Similarity (fuzzy) Search on all repositories; a specific repository must be specified.
- Do not set ReturnCount when using FuzzySearchExpression. It can result in a SOAP fault.

A similarity (fuzzy) score can be specified for a Similarity (fuzzy) search. This score is needed to return a matching record. The value must be between 0 and 1.0. The default is 0.8.

- Standard paging elements such as `<StartCount>` and `<TotalCount>` in the `<ReturnCount>` element (optional). These paging elements help you achieve windowing/pagination of a large result set.

Text Search Fuzzy Query Service with and without Weight

For Text Search Attribute Fuzzy Query web service, you can specify `weight` as an attribute of the `BaseExpression` tag:

- Specifying an attribute set along with weight
  `<ns:BaseExpression attribute="NAME" weight="0.9">Michel</ns:BaseExpression>
  <ns:BaseExpression attribute="CITY" weight="1.0">New York</ns:BaseExpression>`
Specifying an attribute set along without weight

<ns:BaseExpression attribute="NAME">Michel</ns:BaseExpression>
<ns:BaseExpression attribute="CITY">New York</ns:BaseExpression>

Index Entity Fuzzy Query Service

The Index Entity Fuzzy Query web service allows you to perform search on IndexEntity, which is specified in the IndexerConfig.xml. This file is located in the $MQ_HOME/config folder. For more information about this file, refer to TIBCO MDM System Administrator's Guide.

To use Index Entity Fuzzy Query web service, specify the entity name, similarity score, text search fuzzy query service details, and standard paging elements.

Example 16  IndexEntity Request

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
  <soapenv:Header />
  <soapenv:Body>
    <ns:TextSearchRequest>
      <ns:UserInfo>
        <ns:UserName>jsmith</ns:UserName>
        <ns:Password>jsmith</ns:Password>
        <ns:Enterprise>tech</ns:Enterprise>
      </ns:UserInfo>
      <ns:IndexEntityFuzzyQuery>
        <ns:EntityName>Customer_OfficeAddress</ns:EntityName>
        <ns:SimilarityScore>0.6</ns:SimilarityScore>
        <ns:Repository>
          <ns:Name>CUSTOMER</ns:Name>
          <ns:FuzzyQuery>
            <ns:BaseExpression attribute="LASTNAME">Smith</ns:BaseExpression>
          </ns:FuzzyQuery>
          <ns:Relationship>
            <ns:Name>OFFICEADDRESS</ns:Name>
          </ns:Relationship>
        </ns:Repository>
        <ns:Repository>
          <ns:Name>ADDRESS</ns:Name>
        </ns:Repository>
      </ns:IndexEntityFuzzyQuery>
    </ns:TextSearchRequest>
  </soapenv:Body>
</soapenv:Envelope>
Example 17  IndexEntity Response
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns:TextSearchResponse
      xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
      <ns:Records>
        <ns:Record searchScore="1.0" searchScoreString=" ">
          <ns:ExternalKeys>
            <ns:Key name="MASTERCATALOGNAME" type="string">CUSTOMER</ns:Key>
            <ns:Key name="PRODUCTID" type="string">CUST102</ns:Key>
            <ns:Key name="PRODUCTIDEXT" type="string">4</ns:Key>
          </ns:ExternalKeys>
          <ns:EntityData>
            <ns:Attribute name="Gender" type="string">M</ns:Attribute>
            <ns:Attribute name="OWNERID" type="decimal">2147494649</ns:Attribute>
            <ns:Attribute name="CATALOGVERSIONNUMBER" type="decimal">1</ns:Attribute>
            <ns:Attribute name="PRODUCTKEYID" type="decimal">2147494791</ns:Attribute>
            <ns:Attribute name="MODVERSION" type="decimal">1</ns:Attribute>
            <ns:Attribute name="PARENTVERSION" type="decimal">0</ns:Attribute>
            <ns:Attribute name="LastName" type="string">Smith</ns:Attribute>
          </ns:EntityData>
        </ns:Record>
      </ns:Records>
    </ns:TextSearchResponse>
  </soapenv:Body>
</soapenv:Envelope>
<ns:Attribute name="MODMEMBERID" type="decimal">2147485082</ns:Attribute>
<ns:Attribute name="PROCESSLOGID" type="">2148583566</ns:Attribute>
<ns:Attribute name="LASTIMPORTTIME" type="decimal">1.306997374356E12</ns:Attribute>
<ns:Attribute name="ACTIVE" type="string">Y</ns:Attribute>
<ns:Attribute name="MODDATE" type="date">2011-06-02 12:19:48.0</ns:Attribute>
<ns:Attribute name="Firstname" type="string">John</ns:Attribute>
<ns:Attribute name="STATE" type="string">CONFIRMED</ns:Attribute>
<ns:Attribute name="OWNERTYPE" type="decimal">21</ns:Attribute>
<ns:Attribute name="LASTCONFIRMEDVERSION" type="decimal">1</ns:Attribute>
<ns:Attribute name="Age" type="integer">23</ns:Attribute>
<ns:Attribute name="CHECKSUM" type="decimal">-1090762682</ns:Attribute>
<ns:Attribute name="CREATIONDATE" type="date">2011-06-02 12:19:34.0</ns:Attribute>
<ns:Attribute name="DOB" type="date">1986-01-15 00:00:00.0</ns:Attribute>
</ns:EntityData>
<ns:RelationshipData>
<ns:Relationship>
<ns:RelationType>OfficeAddress</ns:RelationType>
<ns:RelatedEntities>
<ns:Record searchScore="1.0" searchScoreString="">
<ns:ExternalKeys>
<ns:Key name="MASTERCATALOGNAME" type="string">ADDRESS</ns:Key>
<ns:Key name="PRODUCTID" type="string">ADDR231</ns:Key>
<ns:Key name="PRODUCTIDEXT" type="string">1</ns:Key>
</ns:ExternalKeys>
<ns:EntityData>
<ns:Attribute name="ADDRESS1" type="string">Gokhale Nagar</ns:Attribute>
<ns:Attribute name="Country" type="string">India</ns:Attribute>
<ns:Attribute name="OWNERID" type="decimal">2147494649</ns:Attribute>
</ns:EntityData>
</ns:Record>
<ns:Attribute name="CATALOGVERSIONNUMBER" type="decimal">1</ns:Attribute>
<ns:Attribute name="PRODUCTKEYID" type="decimal">2147495316</ns:Attribute>
<ns:Attribute name="MODVERSION" type="decimal">1</ns:Attribute>
<ns:Attribute name="PARENTVERSION" type="decimal">0</ns:Attribute>
<ns:Attribute name="MODMEMBERID" type="decimal">2147485082</ns:Attribute>
<ns:Attribute name="LASTIMPORTTIME" type="decimal">1306997374356</ns:Attribute>
<ns:Attribute name="PROCESSLOGID" type="">2148583566</ns:Attribute>
<ns:Attribute name="Zip" type="integer">411016</ns:Attribute>
<ns:Attribute name="ACTIVE" type="string">Y</ns:Attribute>
<ns:Attribute name="MODDATE" type="date">2011-06-02 12:19:47.0</ns:Attribute>
<ns:Attribute name="STATE" type="string">CONFIRMED</ns:Attribute>
<ns:Attribute name="OWNERTYPE" type="decimal">21</ns:Attribute>
<ns:Attribute name="LASTCONFIRMEDVERSION" type="decimal">1</ns:Attribute>
<ns:Attribute name="City" type="string">Pune</ns:Attribute>
<ns:Attribute name="CHECKSUM" type="decimal">-1088085556</ns:Attribute>
<ns:Attribute name="CREATIONDATE" type="date">2011-06-02 12:19:34.0</ns:Attribute>
</ns:EntityData>
</ns:Record>
</ns:RelatedEntities>
</ns:Relationship>
</ns:RelationshipData>
</ns:Record>
</ns:Records>
<ns:ReturnCount>
<ns:StartCount>1</ns:StartCount>
<ns:TotalCount>1</ns:TotalCount>
</ns:ReturnCount>
</ns:TextSearchResponse>
</soapenv:Body>
</soapenv:Envelope>
Get DataSource List Service

This service allows you to get a list of data sources of the currently logged-in users for an enterprise. Only user authorized to access data source can get data source details. You can also retrieve a list of data sources based on filter criteria. For example, if you want to retrieve a data source of the specific user.

For information on authentication of Get DataSource List Service, refer to Table 7, Authentication of Data Web Services, on page 22.

To use this web service:

1. Specify the user details such as user name, password, and enterprise for the user.
2. Invoke the web service by specifying the following endpoint URL:
   
   http://<servername> :<port>/eml/services/DataService/getDatasourceListAction

A sample XML is available in:

$MQ_HOME/schema/DataService/2.0/samples/datasource/GetDataSourceList-request.xml

$MQ_HOME/schema/DataService/2.0/samples/datasource/GetDataSourceList-response.xml

Get DataSource List Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>DatasourceDetails</td>
<td>Details of search criteria and data source name.</td>
</tr>
<tr>
<td>Exact Search</td>
<td>Specify the true or false value.</td>
</tr>
<tr>
<td></td>
<td>• True returns the data sources list exactly matching with the filter criteria.</td>
</tr>
<tr>
<td></td>
<td>• False returns the data sources list that contains the specified filter criteria.</td>
</tr>
</tbody>
</table>
Get DataSource List Service - Outputs

The filtered list of all the data sources is returned for the logged in user. For each data source, the data source ID, its name, description, file name, its format, source table name, delimiter code, and delimiter value, date format, modification member ID, modification date, modification version, active, and data source column details are returned.

For an SQL type data source, the response contains the FileLastUploadedDate element. Also, the TableLastUploadedDate element appears for a normal data source. These are unused elements. In such cases, interpret the response as data last uploaded.

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
Get Repository List Service

This web service allows you to get a list of repositories that can be accessed by the currently logged-in user.

For information on authentication of Get Repository List Service, refer to Table 7, Authentication of Data Web Services, on page 22.

To use this web service:

1. Specify the user details such as user name, password, and enterprise name.
2. Invoke the web service by specifying the endpoint URL as follows:

   http://<servername>:<port>/eml/services/DataService/GetRepositoryListAction

   A list of all repositories is returned in response.

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/GetRepositoryList-request.xml

$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/GetRepositoryList-response.xml

Get Repository List Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Name of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the user.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
</tbody>
</table>

Get Repository List Service - Outputs

A list of repositories that can be accessed by currently logged-in user is returned. For each repository, the name of repository, description, modification member ID, modification date, modification version, and table name are returned. For example:
‘TotalCount’ indicates the total number of repositories that met the search criteria of the request. For example:

<ns:ReturnCount>
  <ns:TotalCount>2</ns:TotalCount>
</ns:ReturnCount>

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
Get Related Repository List Service

This web service allows you to get a list of relationships both forward and reverse for every repository in an enterprise. For the Export Metadata functionality, if you select the parent repository, its related repositories need to be exported together. Using this web service, the icon is displayed on the Export Wizard to identify the related repository. Therefore, while exporting metadata when you click the parent repository in the Available Repositories list, the icon is displayed next to its related repositories. For example, consider a scenario where Customer repository is related to Address repository. If you select the Customer repository for export, the Address repository is also automatically selected for export because a relationship is defined between Customer and address repositories. For more information about selecting repositories for export, refer to Exporting and Importing Metadata chapter in User’s guide.

For information on authentication of Get Related Repository List web service, refer to Table 7, Authentication of Data Web Services, on page 22.

To use this web service:

1. Specify the user details such as user name, password, and enterprise name.
2. Invoke the web service by specifying the endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/DataService/RelatedRepositoryListAction

   A list of all relationships is returned in response.

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/RelatedRepositoryList_Request.xml

$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/RelatedRepositoryList_Response.xml

Get Related Repository List Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Name of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the user.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
</tbody>
</table>
Get Related Repository List Service - Outputs

A list of relationships is returned.
Get Repository with Dependant Artifacts Service

This web service allows you to get a list of all the dependent artifacts for every repository that you have selected during export. For the Export Metadata functionality, if you select the repository, its dependent artifacts need to be exported together. The dependent artifacts include the input maps and output maps. It also includes the dependency of input maps over data sources and dependency of output maps over synchronization formats. For example, consider a scenario where you have selected the Customer repository and its input maps, such as InputMap1 and InputMap2. However, you have not selected the CustDataSource data source on the Select Data Sources screen. In this case, a warning message is displayed to inform that the dependency exist between the input map and a data source. These dependencies are captured by the GetRepositoryWithDependentArtifacts web service. For more information about dependency messages, refer to Exporting and Importing Metadata chapter in User’s guide.

For information on authentication of Get Related Repository List web service, refer to Table 7, Authentication of Data Web Services, on page 22.

To use this web service:

1. Specify the user details such as user name, password, and enterprise name.
2. Invoke the web service by specifying the endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/DataService/RelatedRepositoryListAction

   A list of all relationships is returned in response.

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/RelatedRepositoryList_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/RelatedRepositoryList_Response.xml

Get Repository With Dependant Artifacts Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Name of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the user.</td>
</tr>
</tbody>
</table>
### Get Repository With Dependant Artifacts Service - Outputs

A list of dependant artifacts is returned.

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>Repository Name</td>
<td>Name of the repository.</td>
</tr>
</tbody>
</table>
**Initiate Synchronization Service**

This service allows you to synchronize records on a Trading Partner. This service operates in two modes, Async and Sync.

This service takes Synchronization profile name and repository name as mandatory input and ValidFrom and ValidUntil date as an optional input.

For information on authentication of Initiate Synchronization Service, refer to Table 7, Authentication of Data Web Services, on page 22.

To use this web service:

1. Specify the user name, password, and enterprise in the request criteria.
2. Specify Synchronization profile information such as Name, Repository Name, ValidFrom, and ValidUntil.
3. Specify the execmode as "ASYNCHR" or "SYNCHR".
4. Invoke the webservice by specifying the endpoint URL as follows:
   ```
   http://<servername>:<port>/eml/services/DataService/synchronize
   Action
   ```
   You can synchronize the records in the Async or Sync mode successfully.

A sample XML is available in

- $MQ_HOME/schema/DataService/2.0/samples/repository/record-synchronization/Synchronization-Async-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/repository/record-synchronization/Synchronization-Async-Response.xml
- $MQ_HOME/schema/DataService/2.0/samples/repository/record-synchronization/Synchronization-Sync-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/repository/record-synchronization/Synchronization-Sync-Response.xml

**Initiate Synchronization Service - Inputs**

This web service accepts the following inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>Input</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Name (Mandatory)</td>
<td>Name of the synchronization profile.</td>
</tr>
<tr>
<td>execmode</td>
<td>Specify ASYNCHR or SYNCHR value.</td>
</tr>
<tr>
<td>RepositoryName (Mandatory)</td>
<td>Name of the repository.</td>
</tr>
<tr>
<td>ValidFrom (Optional)</td>
<td>Valid from date.</td>
</tr>
<tr>
<td>ValidUntil (Optional)</td>
<td>Valid until date.</td>
</tr>
</tbody>
</table>

**Initiate Synchronization Service - Outputs**

The response message contains EventId, EventStatus, and EventState of the event. It also contains the DocumentId and DocumentType of the data file.

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
Upload Metadata Service

This service allows you to upload metadata to TIBCO MDM. You can upload jar, zip, or XML file. This functionality is similar to import metadata from TIBCO MDM user interface. You can request the Upload Metadata Service in the SYNC and ASYNC modes.

For information on authentication of Upload Metadata Service, refer to Table 7, Authentication of Data Web Services, on page 22.

To use this web service:

1. Specify the user details such as user name, password, and enterprise for the user.
2. Specify the execmode as ASYNCHR or SYNCHR.
3. Specify upload metadata information such as FileName, FileType, and BinaryData.
4. Invoke the web service by specifying the following endpoint URL:

   http://<servername>:<port>/eml/services/DataService/metadataAction

A sample request XML is available in:

$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/UploadMetadata_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/repository/metadata/UploadMetadata_Response.xml

Upload Metadata Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>execmode</td>
<td>Specify ASYNCHR or SYNCHR value.</td>
</tr>
</tbody>
</table>
Upload Metadata Service - Outputs

If the upload is successful, a success message along with event ID is returned.

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
Extract Data Service

This service allows you to extract data from TIBCO MDM using custom implementation. A request can be sent to extract data from database or other sources, for example, Netrics. Specify the Data provider and Data processor to extract and process data. Specify additional parameters such as SQL query parameters if data is retrieved from database as key value pair.

For more details on implementing the framework, refer to the TIBCO MDM Customization Guide.

For information on authentication of Extract Data Service, refer to Table 7, Authentication of Data Web Services, on page 22.

To use this web service:

1. Specify the user details such as user name, password, and enterprise for the user.
2. Specify the execmode as "ASYNCHR" or "SYNCHR".
3. Specify Extract Details information such as DataProvider, DataProcessor, and BatchSize.
4. Specify parameter information such as key and its value. This is an optional parameter.
5. Invoke the web service by specifying the following endpoint URL:

   http://<servername>:<port>/eml/services/DataService/extractData

A sample request XML is available in:

$MQ_HOME/schema/DataService/2.0/samples/repository/dataextractor/DataExtractor_Request.xml

$MQ_HOME/schema/DataService/2.0/samples/repository/dataextractor/DataExtractor_Response.xml

Extract Data Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
</tbody>
</table>
### Extract Data Service - Outputs

If the spawn is successful, the event ID is returned.

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
Quick Export Service Using Data Extractor

The Quick Export feature uses the data extractor framework that allows you to trigger the export operation using the Extract Data service.

You can export records using the following Quick Export services:

- Data Filter Query
- Related Records
- Subset Data

Quick Export Data Filter Query

You can export golden copy records by specifying catalog ID. The golden copy records are confirmed records. You can also create an SQL query to filter records based on the criteria described in the inputs of the search query.

Quick Export Data Filter Query - Inputs

This web service accepts the following inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
<th>Expected Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataProvider</td>
<td>Fully qualified package name of the IDataProvider interface.</td>
<td>com.tibco.mdm.repository.engine.dataprovider and ProductInfoDataProvider</td>
</tr>
<tr>
<td>DataProcessor</td>
<td>Fully qualified package name of the IDataProcessor interface.</td>
<td>com.tibco.mdm.repository.engine.dataprocessor and RecordToCSVProcessor</td>
</tr>
<tr>
<td>CATALOGID</td>
<td>An ID of the catalog for which quick export needs to be triggered.</td>
<td>User specified</td>
</tr>
</tbody>
</table>
Quick Export Related Records

You can export related records by specifying parent record ID, extension, and relationship name.

**Quick Export Related Records - Inputs**

This web service accepts the following inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
<th>Expected Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEARCHQUERY (Optional)</td>
<td>Searches records based on filter criteria and exports all golden copy records for the repository. The Search Query must select productkeyid and modversion columns. If a table contains different column names, ensure that columns are aliased in the SQL query. For example, to export all confirmed Person repository records with PERSON_NAME attribute starting with 'A': select gc.productkey &quot;productkeyid&quot;, gc.version &quot;modversion&quot;, p.cperson_name from goldencopy gc, person p where gc.productkey=p.cproductkeyid and gc.version=p.cmodversion and p.cperson_name like 'A%'</td>
<td>User specified</td>
</tr>
<tr>
<td>DataProvider</td>
<td>Fully qualified package name of the IDataProvider interface.</td>
<td>com.tibco.mdm.repository.engine.dataprocessing.RelatedRecordsDataProvider</td>
</tr>
<tr>
<td>DataProcessor</td>
<td>Fully qualified package name of the IDataProcessor interface.</td>
<td>com.tibco.mdm.repository.engine.dataprocessing.RecordToCSVProcessor</td>
</tr>
<tr>
<td>PARENTCATALOGID</td>
<td>Catalog ID of parent record whose related records need to be exported.</td>
<td>User specified</td>
</tr>
<tr>
<td>Input</td>
<td>Description</td>
<td>Expected Value</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>PARENTRECORDPRODUCTID (Optional)</td>
<td>Product ID of parent record whose related records need to be exported. Either this parameter or PARENTRECORDPRODUCTKEYID must be available.</td>
<td>User specified</td>
</tr>
<tr>
<td>PARENTRECORDEXTENSION (Optional)</td>
<td>Product ID of parent record whose related records need to be exported.</td>
<td>User specified</td>
</tr>
<tr>
<td>PARENTRECORDM MODVERSION</td>
<td>Version of parent record whose related records need to be exported.</td>
<td>User specified</td>
</tr>
<tr>
<td>PARENTRECORDPRODUCTKEYID (Optional)</td>
<td>Product key ID of parent record whose related records need to be exported. Either this parameter or PARENTRECORDPRODUCTID must be available.</td>
<td>User specified</td>
</tr>
<tr>
<td>RELATIONSHIPNAME</td>
<td>Specifies name of the relationship for the records. For example, PersonToAddress relationship related to the Person and Address repositories. In this case, the records related to PersonToAddress relationship are exported.</td>
<td>User specified</td>
</tr>
<tr>
<td>EVENTID (Optional)</td>
<td>Specifies parent record as draft</td>
<td>User specified</td>
</tr>
</tbody>
</table>
Quick Export Subset Data

You can export subset data by specifying subset name and catalog ID.

Quick Export Subset Data - Inputs

This web service accepts the following inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
<th>Expected Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataProvider</td>
<td>Fully qualified package name of the IDataProvider interface.</td>
<td>com.tibco.mdm.repository.engine.dataprovider.SubsetDataProvider</td>
</tr>
<tr>
<td>DataProcessor</td>
<td>Fully qualified package name of the IDataProcessor interface.</td>
<td>com.tibco.mdm.repository.engine.dataprocessor.RecordToCSVProcessor</td>
</tr>
<tr>
<td>SUBSETNAME (Optional)</td>
<td>Name of the subset. Either this parameter or SUBSETID must be available.</td>
<td>User specified</td>
</tr>
<tr>
<td>CATALOGID (Optional)</td>
<td>Catalog ID for which subset is created. If a subset name is provided, the Catalog ID must be available.</td>
<td>User specified</td>
</tr>
<tr>
<td>SUBSETID (Optional)</td>
<td>An ID of the subset whose data are exported. Either this parameter or the subset name must be available.</td>
<td>User specified</td>
</tr>
</tbody>
</table>

Quick Export Common Parameters

Quick Export data extractor web services support the following common parameters:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
<th>Expected Value</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELIMITER (Optional)</td>
<td>Refers to the delimiter for data in a file. The delimiter must be a single character.</td>
<td>User specified</td>
<td>Comma (,)</td>
</tr>
<tr>
<td>TEXTQUALIFIER (Optional)</td>
<td>The qualifier used to differentiate the field data from the delimiter.</td>
<td>User specified</td>
<td>Double Quotes (&quot;)</td>
</tr>
<tr>
<td>Input</td>
<td>Description</td>
<td>Expected Value</td>
<td>Default Value</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>FILEEXTENSION (Optional)</td>
<td>An extension of the exported output data files.</td>
<td>User specified</td>
<td>CSV</td>
</tr>
<tr>
<td>EMAILDELIVERY (Optional)</td>
<td>Specifies whether an email to be sent at the end of the operation.</td>
<td>True or False</td>
<td>False</td>
</tr>
<tr>
<td>EXPORTATTRIBUTES (Optional)</td>
<td>Specifies whether to export all or displayable attributes.</td>
<td>All or displayable (case sensitive)</td>
<td>All</td>
</tr>
<tr>
<td>EMAILTEMPLATE (Optional)</td>
<td>Template used for sending email notification containing the File Download link. Note that email template must have an anchor element with downloadUrl ID. The application sets the download link for this anchor.</td>
<td>Fully classified class name of the email template</td>
<td>com.tibco.mdm.ui.workflow.engine.emailtemplates&gt;EmailQuickExport</td>
</tr>
</tbody>
</table>
GetClassificationSchemeList

This web service allows you to retrieve a list of classification schemes for a particular repository.

To use this web service:

1. Specify the user name, password, and enterprise in the request criteria.
2. Specify information such as Repository Name.
3. Invoke the web service by specifying the Endpoint URL as follows:

   http://<servername>:<port>/eml/services/DataService/GetClassificationSchemeListAction

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/repository/classification/
GetClassificationSchemeListRequest.xml

$MQ_HOME/schema/DataService/2.0/samples/repository/classification/
GetClassificationSchemeListResponse.xml

GetClassificationSchemeList - Inputs

This web service accepts the following inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Specify the Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Specify the password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Specify the name of the enterprise.</td>
</tr>
<tr>
<td>RepositoryName</td>
<td>Specify the name of the repository. (Mandatory)</td>
</tr>
<tr>
<td>ReturnCount</td>
<td>Specify the return count for paginated response. You must specify the 'startCount' and 'totalCount' of the schemes required in the request. This field is optional.</td>
</tr>
</tbody>
</table>

GetClassificationSchemeList - Outputs

The response message contains Classification Scheme ID, Classification Scheme Name, Classification Scheme Type, and Classification Scheme Description.
Request:

```xml
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
<soapenv:Header/>
<soapenv:Body>
  <ClassificationSchemeListRequest
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
    <UserInfo>
      <UserName>a</UserName>
      <Password>a</Password>
      <Enterprise>a</Enterprise>
    </UserInfo>
    <Repository>FBT_Repository</Repository>
  </ClassificationSchemeListRequest>
</soapenv:Body>
</soapenv:Envelope>
```

Response:

```xml
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
<soapenv:Header/>
<soapenv:Body>
  <ClassificationSchemeListResponse
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
    <ClassificationScheme>
      <Id>12101</Id>
      <Name>FBT</Name>
      <Description>FBT Desc</Description>
      <Type>AttributeBased</Type>
      <ModDate>12/12/12</ModDate>
      <Modversion>1</Modversion>
      <Active>Y</Active>
    </ClassificationScheme>
    <ClassificationScheme>
      <Id>12105</Id>
      <Name>Electronics</Name>
      <Description>Electronics Desc</Description>
      <Type>Rulebase</Type>
      <ModDate>12/12/12</ModDate>
  </ClassificationSchemeListResponse>
</soapenv:Body>
</soapenv:Envelope>
```
<Modversion>1</Modversion>
<ModMemberId>1</ModMemberId>
<Active>Y</Active>
</ClassificationScheme>
</ClassificationSchemeListResponse>
</soapenv:Body>
</soapenv:Envelope>

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
GetClassificationSchemeAttributeList

This service allows you to retrieve all details of the classification scheme including attributes (for attribute-based scheme), and rulebase details (for rulebase-based scheme).

This service takes the name of the Repository name, classification scheme name, and classification scheme ID as input and returns all the details of the classification scheme including attributes (for attribute-based scheme), and rulebase details (for rulebase-based scheme).

To use this web service:

1. Specify the user name, password, and enterprise in the request criteria.
2. Specify information such as Repository Name, and Classification Scheme Name.
3. Invoke the web service by specifying the Endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/DataService/GetClassificationSchemeAttributeListAction

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/repository/classification/getClassificationSchemeAttrList_Request.xml

$MQ_HOME/schema/DataService/2.0/samples/repository/classification/getClassificationSchemeAttrList_Response.xml

GetClassificationSchemeAttributeList - Inputs

This web service accepts the following inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Specify the login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Specify the password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Specify the name of the enterprise.</td>
</tr>
<tr>
<td>RepositoryName (Mandatory)</td>
<td>Specify the name of the repository.</td>
</tr>
<tr>
<td>ClassificationScheme ID</td>
<td>Specify the ID of the classification scheme.</td>
</tr>
</tbody>
</table>
### GetClassificationSchemeAttributeList - Outputs

The response message contains Classification Scheme ID, Classification Scheme Name, Classification Scheme Type, Classification Scheme Description, and Classification Scheme Attributes.

**Request:**

```xml
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
  <soapenv:Header/>
  <soapenv:Body>
    <ClassificationSchemeAttributeListRequest
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
      <UserInfo>
        <UserName>a</UserName>
        <Password>a</Password>
        <Enterprise>a</Enterprise>
      </UserInfo>
      <Repository>
        <Name>FBT_Repository</Name>
        <ClassificationScheme>
          <Id>FBT</Id>
          <Name>FBT</Name>
        </ClassificationScheme>
      </Repository>
    </ClassificationSchemeAttributeListRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

**Response:**

```xml
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
  <soapenv:Header/>
  <soapenv:Body>

  </soapenv:Body>
</soapenv:Envelope>
```
For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
GetClassificationCodeHierarchy

Using the GetClassificationCodeHierarchy web service, you can retrieve the classification code hierarchy depending on the context classification codes provided in the request.

The request can contain a single classification code or a hierarchy of codes. The request is provided in two ways:

- Code Hierarchical Notation
- TreePath Notation

The web service also supports pagination of response. Specify the startCount and totalCount of the classification codes required in the request. The startCount is optional. This determines the total count of code which is returned in the response.

While using TreePath Notation, the name of the code is not required, only the classification Code must be specified. The response is always in the ClassificationCode Notation. For more information, refer the samples, Sample 1: GetClassification Code from Level 1 using treepath Notation, on page 218

To use the web service:

1. Specify the user name, password, and enterprise in the request criteria.
2. Specify information such as Repository Name, and Classification Scheme Name.
3. Invoke the web service by specifying the Endpoint URL as follows:
   http://<servername>:<port>/eml/services/DataService/.....Action

A sample XML is available in

$MQ_HOME/schema/DataService/2.0/samples/repository/classification/GetClassificationCode_fromRoot_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/repository/classification/GetClassificationCode_fromRoot_Response.xml
$MQ_HOME/schema/DataService/2.0/samples/repository/classification/GetClassificationCode_fromLevel1_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/repository/classification/GetClassificationCode_fromLevel1_Response.xml
$MQ_HOME/schema/DataService/2.0/samples/repository/classification/GetClassificationCode_withmultipleCodes_Request.xml
$MQ_HOME/schema/DataService/2.0/samples/repository/classification/GetClassificationCode_withmultipleCodes_Response.xml
GetClassificationCodeHierarchy - Inputs

This web service accepts the following inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Specify the login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Specify the password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Specify the name of the enterprise.</td>
</tr>
<tr>
<td>RepositoryName (Mandatory)</td>
<td>Specify the name of the repository.</td>
</tr>
<tr>
<td>Classification ID</td>
<td>Specify the ID of the classification scheme.</td>
</tr>
<tr>
<td>Classification Name (Mandatory)</td>
<td>Specify the name of the classification scheme.</td>
</tr>
<tr>
<td>FetchDepth</td>
<td>Specify the depth of the response code. Default depth is 1.</td>
</tr>
</tbody>
</table>

Hierarchy Notation

| Code                         | Specify the classification code.                                           |
| Name                         | Specify the classification code name.                                      |
| Level                        | Specify the level in the hierarchy. You need to specify the level from the root node. The root node is same as classification scheme name and its level is 0. |

Treepath Notation

| Level                        | Specify the level in the hierarchy. You need to specify the level from the root node. The root node is same as classification scheme name and its level is 0. |
| Path                         | Specify the code.                                                          |
### GetClassificationCodeHierarchy - Outputs

The response message contains Classification Code, Classification Code Name, Classification Code Description, Classification Code Attributes(if any) and its level.

#### Sample 1: GetClassification Code from Level 1 using treepath Notation

**Request:**

```xml
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
  <soapenv:Header />
  <soapenv:Body>
    <ns:ClassificationCodeRequest>
      <ns:UserInfo>
        <ns:UserName>a</ns:UserName>
        <ns:Password>abc</ns:Password>
        <ns:Enterprise>ws</ns:Enterprise>
      </ns:UserInfo>
      <ns:Repository>
        <ns:Name>FBT</ns:Name>
        <ns:ClassificationScheme>
          <ns:Id>236901</ns:Id>
          <ns:Name>FBT-Scheme</ns:Name>
        </ns:ClassificationScheme>
      </ns:Repository>
      <ns:ClassificationCodeHierarchy>
        <ns:FetchDepth>1</ns:FetchDepth>
        <ns:ClassificationCodeTreePath>
          ReturnCount Specify the return count for the paginated response.
          You must specify the startCount and totalCount of the codes required in the request. This field is optional.

          **Note:** If ReturnCount is specified, the value of FetchDepth is ignored.
```
GetClassificationCodeHierarchy

Response:

<soapenv:Body>

<ns:ClassificationCodeResponse>

<ns:ClassificationCode Level="2">
  <ns:Code>50181700</ns:Code>
  <ns:Name>BAKING MIXES</ns:Name>
  <ns:Description>Baking Mixes</ns:Description>
</ns:ClassificationCode>

<ns:ClassificationCode Level="2">
  <ns:Code>50181900</ns:Code>
  <ns:Name>BREAD</ns:Name>
  <ns:Description>Bread</ns:Description>
</ns:ClassificationCode>

<ns:ClassificationCode Level="2">
  <ns:Code>50182000</ns:Code>
  <ns:Name>SWEET BAKERY PRODUCTS</ns:Name>
  <ns:Description>Sweet Bakery Products</ns:Description>
</ns:ClassificationCode>

<ns:ClassificationCode Level="2">
  <ns:Code>50182100</ns:Code>
  <ns:Name>BISCUITS/COOKIES</ns:Name>
  <ns:Description>Biscuits/Cookies</ns:Description>
</ns:ClassificationCode>

<ns:ClassificationCode Level="2">
  <ns:Code>50182200</ns:Code>
  <ns:Name>SAVOURY BAKERY PRODUCTS</ns:Name>
  <ns:Description>Savoury Bakery Products</ns:Description>
</ns:ClassificationCode>

</ns:ClassificationCodeResponse>
</soapenv:Body>
Sample 2: GetClassifcationCode from Root using Hierarchical Notation

Request:
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
  <soapenv:Header />
  <soapenv:Body>
    <ns:ClassificationCodeRequest>
      <ns:UserInfo>
        <ns:UserName>a</ns:UserName>
        <ns:Password>abc</ns:Password>
        <ns:Enterprise>ws</ns:Enterprise>
      </ns:UserInfo>
      <ns:Repository>
        <ns:Name>FBT</ns:Name>
        <ns:ClassificationScheme>
          <ns:Id>236901</ns:Id>
          <ns:Name>FBT-Scheme</ns:Name>
        </ns:ClassificationScheme>
      </ns:Repository>
      <ns:ClassificationCodeHierarchy>
        <ns:FetchDepth>1</ns:FetchDepth>
        <!-- You have a CHOICE of the next 2 items at this level. i.e
          - choice of using Hierarchical Notation or TreePath Notation -->
        <ns:ClassificationCodeTreePath>
          <ns:TreePath>
            <ns:Path level="0">FBT-SCHEME</ns:Path>
          </ns:TreePath>
        </ns:ClassificationCodeTreePath> -->
      </ns:ClassificationCodeHierarchy>
    </ns:ClassificationCodeRequest>
  </soapenv:Body>
</soapenv:Envelope>
<ns:ClassificationCode Level="0">
  <ns:Code>FBT-SCHEME</ns:Code>
  <ns:Name>FBT-SCHEME</ns:Name>
</ns:ClassificationCode>
</ns:ClassificationCodes>
</ns:ClassificationCodeHierarchy>
<!-- The next element ‘ReturnCount’ is optional. Use only if pagination is desired -->
<ns:ReturnCount>
  <ns:StartCount>1</ns:StartCount>
  <ns:TotalCount>5</ns:TotalCount>
</ns:ReturnCount>
</ns:ClassificationCodeRequest>
</soapenv:Body>
</soapenv:Envelope >

Response:
<soapenv:Envelope xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/
 xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsd1/2.0">
  <soapenv:Body>
    <ns:ClassificationCodeResponse>
      <ns:ClassificationCode Level="1">
        <ns:Code>50200000</ns:Code>
        <ns:Name>BEVERAGES</ns:Name>
        <ns:Description>Beverages</ns:Description>
      </ns:ClassificationCode>
      <ns:ClassificationCode Level="1">
        <ns:Code>50180000</ns:Code>
        <ns:Name>BREAD/BAKERY PRODUCTS</ns:Name>
        <ns:Description>Bread/Bakery Products</ns:Description>
      </ns:ClassificationCode>
    </ns:ClassificationCodeResponse>
  </soapenv:Body>
</soapenv:Envelope>
Sample 3: GetClassificationCode with Multiple codes in Request

Request:

```xml
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap-envelope/
xmlns:ns="http://www.tibco.com/cim/services/dataservice/wsdl/2.0">
  <soapenv:Header />
  <soapenv:Body>
    <ns:ClassificationCodeRequest>
      <ns:UserInfo>
        <ns:UserName>a</ns:UserName>
        <ns:Password>abc</ns:Password>
        <ns:Enterprise>ws</ns:Enterprise>
      </ns:UserInfo>
      <ns:Repository>
        <ns:Name>FBT</ns:Name>
        <ns:ClassificationScheme>
          <ns:Id>236901</ns:Id>
          <ns:Name>FBT-Scheme</ns:Name>
        </ns:ClassificationScheme>
      </ns:Repository>
      <ns:ClassificationCodeHierarchy>
        <ns:FetchDepth>1</ns:FetchDepth>
        <ns:ClassificationCodes>
          <ns:ClassificationCode>
            <ns:ClassificationCode Level="1">
              <ns:Code>50230000</ns:Code>
              <ns:Name>FOOD/BEVERAGE/TOBACCO PACKS</ns:Name>
              <ns:Description>Food/Beverage/Tobacco Packs</ns:Description>
            </ns:ClassificationCode>
          </ns:ClassificationCode>
          <ns:ClassificationCode>
            <ns:ClassificationCode Level="1">
              <ns:Code>50250000</ns:Code>
              <ns:Name>FRUITS (FRESH)</ns:Name>
              <ns:Description>Fruits (Fresh)</ns:Description>
            </ns:ClassificationCode>
          </ns:ClassificationCode>
        </ns:ClassificationCodes>
      </ns:ClassificationCodeHierarchy>
    </ns:ClassificationCodeRequest>
</soapenv:Body>
</soapenv:Envelope>
```
<ns:ClassificationCode Level="1">
  <ns:Code>50180000</ns:Code>
  <ns:Name>BREAD/BAKERY PRODUCTS</ns:Name>
  <ns:ClassificationCode Level="2">
    <ns:Code>50181700</ns:Code>
    <ns:Name>BAKING/COOKING MIXES/SUPPLIES</ns:Name>
    </ns:ClassificationCode>
  </ns:ClassificationCode>
</ns:ClassificationCodes>

For information on failure messages, refer to Table 12, Faults of Data Web Services, on page 28.
Chapter 6  DataLoad Services

This chapter describes the usage and customization of data load web services.

Topics

- Get DataSource Info Service, page 226
- Upload DataSource Service, page 228
- Import Records Service, page 230
- Load Import Records Service, page 232
Get DataSource Info Service

Using this service, you can get the detailed information of a data source for the currently logged-in users. The users who can access the data source can get the data source information.

For information on authentication of Get DataSource Info Service, refer to Table 8, Authentication of DataLoad Web Services, on page 23.

To use this web service:
1. Specify the user details such as user name, password, and enterprise.
2. Specify the name of a datasource.
3. Invoke the web service by specifying the following endpoint URL:
   
   http://<servername>:<port>/eml/services/DataLoadService/getDataSourceInfoAction

A sample XML is available in:

$MQ_HOME/schema/DataService/2.0/samples/datasource/GetDataSourceInfo-request.xml

$MQ_HOME/schema/DataService/2.0/samples/datasource/GetDataSourceInfo-response.xml

Get DataSource Info Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>DataSourceName</td>
<td>Name of the data source.</td>
</tr>
</tbody>
</table>

Get DataSource Info Service - Outputs

For each data source, the following information are returned:

- Data source ID
- Data source name and its description
- File name and its format, source table name
- Delimiter code and value
- Header line count
- Use titles flag
- File last uploaded date
- Table last uploaded date
- Loaded row count
- Text qualifier
- Thousand separator
- Date format
- Modification member ID, modification date, and modification version
- Active
- Transport protocol

For information on failure messages, refer to Table 13, Faults of DataLoad Web Services, on page 29.
Upload DataSource Service

Using this web service, you can upload a data source. To execute this web service, ensure that the data source exists.

To use this web service:

1. Specify user details such as, the user name, password, and enterprise.
2. Specify the data source details such as, the data source name, execution mode, and the file name.
3. Attach the data source file.
4. Invoke the web service by specifying the following endpoint URL:
   
   http://<servername>:<port>/eml/services/DataLoadService/DataSourceUploadAction

   The binary data value is automatically created.

The following are sample XML files:

- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-Async-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-Async-Response.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-Sync-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-Sync-Response.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-DBloaderSync-Request-fail.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-DBloaderSync-Response-fail.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-NonexistingRepository-Request-fail.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadDataSource-NonexistingRepository-Response-fail.xml

Upload DataSource Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>The login ID of the user</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the specified user name</td>
</tr>
</tbody>
</table>
Upload DataSource Service - Outputs

If you upload the data source successfully, the Datasource uploaded successfully message is displayed. The response contains event details depending on the specified execution mode. The event details include the event ID, status, state, and description of the event state and status.

For information on failure messages, refer to Table 13, Faults of DataLoad Web Services, on page 29.

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise</td>
<td>The name of the enterprise</td>
</tr>
<tr>
<td>execmode (Mandatory)</td>
<td>Specify the execution mode. The valid values are ASYNCHR or SYNCHR.</td>
</tr>
<tr>
<td>DataSourceName (Mandatory)</td>
<td>The name of the data source</td>
</tr>
<tr>
<td>FileName</td>
<td>The name of the file. The filename must contain an extension. The extension can be .csv or .txt. Optional for SQL data sources, otherwise mandatory.</td>
</tr>
<tr>
<td>File as attachment</td>
<td>The data source file you can attach using SOAP UI. The file must include the records to be imported. Optional for SQL data sources, otherwise mandatory.</td>
</tr>
<tr>
<td>BinaryData</td>
<td>When you create a new request using SOAP UI, the value for the &lt;BinaryData&gt; parameter is automatically created. This is a unique value for every request, and if the request contains an attachment, specifying the binary data parameter is mandatory.</td>
</tr>
</tbody>
</table>
Import Records Service

Using this web service, you can import records into the repository. To execute this web service, ensure that a repository and an input map exist. Based on the input map options, you can perform the following types of import:

- Database loader
- Relationships only

You can import records based on the values selected in an input map. The following features are considered while importing:

- Sequencing options for concurrent jobs
- Message priorities as configured

To use this web service:

1. Specify user details such as, the user name, password, and enterprise.
2. Specify import record details such as, the execution mode, repository name, and an input map name.
3. Invoke the web service by specifying the following endpoint URL:
   
   http://<servername>:<port>/eml/services/DataLoadService/ImportAction

The following are sample XML files:

- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/ImportRecords-Async-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/ImportRecords-Async-Response.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/ImportRecords-Sync-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/ImportRecords-Sync-Response.xml

Import Records Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>The login ID of the user</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the specified user name</td>
</tr>
<tr>
<td>Enterprise</td>
<td>The name of an enterprise</td>
</tr>
</tbody>
</table>
## Import Records Service - Outputs

The response contains event details based on the specified execution mode. The event details include the event ID, its status and state, and event status and state description.

- For the `ASYNCHR` execution mode, if the records are imported successfully, the **Import initiated successfully** message is displayed.
- For the `SYNCHR` execution mode, if the records are imported successfully, the **Request Completed** message is displayed.

For information on failure messages, refer to Table 13, Faults of DataLoad Web Services, on page 29.
Load Import Records Service

Using this web service, you can upload the data source and import records into the repository. You must have the security permissions to import records. Use this web service when only one data source is associated with an input map. To execute this web service, ensure that a repository, data source, and an input map exist.

To use this web service:
1. Specify user details such as, the user name, password, and enterprise.
2. Specify load import record details such as, the execution mode, repository name, input map name, and a file name.
3. Invoke the web service by specifying the following endpoint URL:

   http://servername:<port>/eml/services/DataLoadService/LoadImportAction

The following are sample XML files:

- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadImportRecords-Sync-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadImportRecords-Sync-Response.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadImportRecords-Async-Request.xml
- $MQ_HOME/schema/DataService/2.0/samples/dataloadservice/LoadImportRecords-Async-Response.xml

Load Import Records Service - Inputs

The following inputs are required for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>The login ID of the user</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the specified user name</td>
</tr>
<tr>
<td>Enterprise</td>
<td>The name of the enterprise</td>
</tr>
<tr>
<td>execmode</td>
<td>Specify the execution mode. The valid values are SYNCHR or ASYNCHR</td>
</tr>
</tbody>
</table>

(Mandatory)
**Load Import Records Service - Outputs**

If you upload the data source and import records successfully, the *Import initiated successfully* message is displayed. The response contains the following:

- The import event details such as, the event ID, status, state, and description of the event state and status. These details are displayed depending on the specified execution mode.
- The import activity summary.
- The number of records processed during the import.

For information on failure messages, refer to Table 13, *Faults of DataLoad Web Services*, on page 29.

### Input Description

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository (Mandatory)</td>
<td>The name of the repository. Specify any existing repository name.</td>
</tr>
<tr>
<td>InputMapName (Mandatory)</td>
<td>The name of an input map for importing data. Any existing input map within the specified repository</td>
</tr>
<tr>
<td>FileName</td>
<td>The name of the data source file. You can attach the file using SOAP UI. The filename must contain the .csv or .txt extension. Optional for SQL data sources, otherwise mandatory.</td>
</tr>
<tr>
<td>BinaryData</td>
<td>When you create a new request using SOAP UI, the value for the <em>&lt;BinaryData&gt;</em> parameter is automatically created. This is a unique value for every request, and if the request contains an attachment, specifying the binary data parameter is mandatory.</td>
</tr>
</tbody>
</table>
Chapter 7  Content Services

This chapter describes the usage and customization of content services.

Topics

- Get Content Service, page 236
Get Content Service

This service allows you to download a file for the specified context. This service takes DocumentId or DocumentPath as an input, also the context from where the file needs to download is specified. If the context is accessible for you, only then you can download the file. Currently, only the Event context is supported.

For information on authentication of Get Content Service, refer to Table 9, Authentication of Content Web Services, on page 24.

To use this web service:

1. Specify user credentials to log on to TIBCO MDM.
2. Specify the Context to which the file belongs. For example, Event. If the context specified is Event, you must have the View Event permission.
3. Specify either the DocumentId or DocumentPath of the file that is to be downloaded.
4. Invoke the webservice by specifying the endpoint URL as follows:

   http://<servername>:<port>/eml/services/ContentService/download

   The file is successfully downloaded.

A sample XML is available in:

$MQ_HOME/schema/DataService/2.0/samples/content/ContentService-Download-Request.xml
$MQ_HOME/schema/DataService/2.0/samples/content/ContentService-Download-Response.xml

Get Content Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Login ID of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the specified user name.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>DocumentId</td>
<td>The ID of the file that is to be downloaded. Either the ID or path must be specified.</td>
</tr>
</tbody>
</table>
## Get Content Service - Outputs

If the file is retrieved successfully, the response contains the file name, file type, inline binary data, and the success code.

For information on failure messages, refer to Table 14, Faults of Content Web Services, on page 31.
Chapter 8  Event Management Services

This chapter describes the usage and customization of event management services.

Topics

- Get Event Info Service, page 240
- Get Event Details Service, page 242
- Get Import Event Status Service, page 244
Get Event Info Service

This service allows you to get event information for a specified event ID.

For information on authentication of Get Event Info Service, refer to Table 10, Authentication of Event Management Web Services, on page 25.

To use this web service:
1. Specify the user name, password, and enterprise in the request criteria.
2. Specify the EventId for whom the event status and event summary needs to be retrieved.
   If the EventId does not exist, an error message is returned in the response.
3. Invoke the web service by specifying the endpoint URL as follows:
   http://<servername>:<port>/eml/services/EventMgtService/getEventInfoAction

A sample XML is available in:
$MQ_HOME/schema/DataService/2.0/samples/eventmgmt/Event-GetInfo-Request.xml
$MQ_HOME/schema/DataService/2.0/samples/eventmgmt/Event-GetInfo-Response.xml

Get Event Info Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Name of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the user.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>EventId</td>
<td>ID of the event for which details need to be retrieved.</td>
</tr>
</tbody>
</table>
Get Event Info Service - Outputs

The response message contains the Event type, Event State, Event Status, and Event Summary. The Event Summary is different for each event type.

The event information returned depends on the type of the event, that is, the event summary information is returned only for the events that have summary. Also, the number of attributes in the summary information varies based on the event. A file attribute in the summary information path or a documentId for that attribute is also returned so that the Content service can be used to download the document using this information.

For information on failure messages, refer to Table 15, Faults of Event Management Web Services, on page 32.
Get Event Details Service

This service allows you to retrieve the event details for the specified Event ID. The event details as displayed in the Event Log screen are retrieved.

For information on authentication of Get Event Details Service, refer to Table 10, Authentication of Event Management Web Services, on page 25.

To use this web service:

1. Specify the user name, password, enterprise for the user, and an event ID.
2. Invoke the web service by specifying the endpoint URL as follows:

   http://<servername>:<port>/eml/services/EventMgtService/GetEventDetailsAction

   A sample XML is available in:

   $MQ_HOME/schema/DataService/2.0/samples/eventmgmt/GetEventDetails_Request.xml
   $MQ_HOME/schema/DataService/2.0/samples/eventmgmt/GetEventDetails_Response.xml

Get Event Details Service - Inputs

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Name of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the user.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>EventId</td>
<td>ID of the event for which details need to be retrieved. This should be a valid event ID.</td>
</tr>
</tbody>
</table>
Get Event Details Service - Outputs

The event details returned depend on the type of the event. For example, the event summary information is returned only for the events that have a summary. If there is a file attribute in the summary information path, the document ID for the attribute is returned. The user uses this document ID to download the document using the contentService.

The response message contains Event Details, Process Details, Process Logs Details, and associated events.

For information on failure messages, refer to Table 15, Faults of Event Management Web Services, on page 32.
**Get Import Event Status Service**

This service allows you to view the status of an import event. To check the status, specify the Event ID of an import event. Before you run this web service, execute the Import Records web service. For information, refer to Import Records Service on page 230.

For information on authentication of Get Import Event Status Service, refer to Table 10, Authentication of Event Management Web Services, on page 25.

To use this web service:

1. Specify the user name, password, enterprise for the user, and an event ID.
2. Invoke the web service by specifying the endpoint URL as follows:
   
   http://<servername>:<port>/eml/services/EventMgtService/GetImportEventStatusAction

   A sample XML is available in:
   
   $MQ_HOME/schema/DataService/2.0/samples/eventmgmt/GetImportEventStatus_Request.xml
   $MQ_HOME/schema/DataService/2.0/samples/eventmgmt/GetImportEventStatus_Response.xml

**Get Import Event Status Service - Inputs**

The following inputs are mandatory for this web service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>Name of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Password of the user.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Name of the enterprise.</td>
</tr>
<tr>
<td>EventId</td>
<td>ID of the event for which details need to be retrieved. This should be a valid event ID.</td>
</tr>
</tbody>
</table>
Get Import Event Status Service - Outputs

The response contains the following:

- Import summary with the Success status.
- Summary of the each activity of Import workflow.
- For a Single repository, the following parameters are displayed:
  RowsProcessed, NewRecords, ModifiedRecords, RejectedRecords,
  EmptyIgnoredRecords, DuplicateRecords show the appropriate values based
  on the Import approval option and the number of records imported.
- For cross-repository, the following parameters are displayed in the
  Relationship Summary: RecordsProcessed, RelationshipCreated,
  RejectedRecords, DuplicateRelationship, RelationshipFailed. They contain the
  appropriate values according to the number of records imported.

For information on failure messages, refer to Table 15, Faults of Event
Management Web Services, on page 32.
Index

A
Administrator Web Services 151
  Create Enterprise Service 157
  Create User Service 161
  Delete User Service 174
  Get Enterprise List Service 159
  Get Entitlement Service 154
  Get Role List Service 178
  Get User List Service 176
  Login Service 152
  Logout Service 153
ASYNCHR mode of web service execution 3
AXIS2 19

C
Content Web Services 235
  Get Content Service 236
customer support xiv, xiv

D
Data Web Services 181, 225
  Extract Data Service 203
  Get DataSource List Service 191, 226
  Get Repository List Service 193
  Initiate Synchronization Service 199
  Text Search Service 182
    Index Entity Fuzzy Query Service 187
    Text Search Exact Query Service 185
    Text Search Fuzzy Query Service 186
  Upload Metadata Service 201

E
Event Management Web Services
  Get Event Details Service 242, 244
  Get Event Service 240
execution mode
  Asynchronous mode (ASYNCHR) 3
  Synchronous mode (SYNCHR) 2

P
Password Encryption and Decryption 33

R
Record Web Services 47
  Initiate Workflow Service 109
  Record Add Modify Services 96
  Record Query MetaData 84
  Record Query Service 61
  Record Query ValidValues (Init) Service 88
  Record Validate Service 94

S
Single Sign-On 18
support, contacting xiv, xiv
SYNCHR mode of web service execution 2
T

technical support  xiv, xiv

W

WebServices - Message Structure  5
Work Item Web Services  117
  Get Custom Work Item Summary Service  142
  Get Predefined Work Item Summary Service  148
  Get Work Item Service  134
  Work Item Lock Service  130
  Work Item Modify Close Service  126
  Work Item Query Parameters Service  118
  Work Item Reassign Service  140
  Work Item UnLock Service  132