



# **`Agile Product Lifecycle Management**

Agile PLM Web Services User Guide

v9.3.1

Part No. E16504-01

September 2010

# Oracle Copyright

*Copyright © 1995, 2010, Oracle and/or its affiliates. All rights reserved.*

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

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

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

## U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

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

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

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third party content, products and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third party content, products or services. The RMW product includes software developed by the Visigoth Software Society.

# CONTENTS

---

Oracle Copyright.....	ii
<b>Introduction to Agile PLM Web Services .....</b>	<b>1</b>
About Service Oriented Architecture (SOA) .....	1
About Web Services .....	1
Core Technologies .....	2
Web Services Description Language (WSDL) .....	2
XML and XML Schema.....	3
Simple Object Access Protocol (SOAP) .....	3
Web Services Architecture .....	3
About Agile PLM Web Services .....	4
Agile PLM Core Web Services .....	4
Agile PLM EC Services .....	5
Agile Recipe & Material Workspace Web Services .....	5
Agile PLM Web Service Authentication and Performance .....	5
Impact on Existing Agile PLM Extensions and Services .....	6
Casual User Interface Integration Examples.....	6
User Interface Integration - MS Word .....	7
User Interface Integration - MS Excel.....	8
User Interface Integration - Portals and Agile Web Client.....	8
User Interface Integration - Mobile ADF .....	8
CAD Integration through EC Services .....	9
Building Casual User Interfaces .....	10
Developing User Interfaces for MS Office .....	10
Developing User Interfaces for Oracle WebCenter and ADF .....	11
<b>Getting Started with Agile Web Services .....</b>	<b>13</b>
Before Building a Web Services Client .....	13
Operational Environment.....	14
Standards Compliance .....	14
Web Services Engines .....	14
Generating and Initializing the Stubs .....	15
Generating Agile Stubs .....	15
Initializing the Client Stubs .....	15
Understanding the MessageElement .....	15
Obtaining the API Names and Attribute IDs .....	16
Special Handling of MessageElements .....	16

Unit of Measure .....	16
Multilist and List .....	16
Money .....	17
Date .....	18
User/Supplier/Customer/Analyst .....	18
Agile Attributes without API Names.....	18
Item Constants .....	19
User Constants .....	19
Understanding the Web Services Responses.....	19
Response Status Code .....	19
Exceptions and Warnings.....	20
Working with Warnings.....	21
<b>Working with Business Objects .....</b>	<b>23</b>
Getting an Object.....	23
Special Handling in the getObject Operation .....	24
Creating an Object .....	24
Saving As a New Object.....	25
Special Handling in the saveAsObject Operation.....	26
Deleting and Undeleting an Object.....	27
Checking the Delete Status .....	28
Updating an Object.....	28
Getting the Status of an Object .....	28
Getting the AutoNumbers.....	29
Getting all the Classes .....	29
Getting the Subclasses of a Class .....	29
Getting Agile Classes .....	30
<b>Subscribing to Agile PLM Objects .....</b>	<b>31</b>
Subscribe Privilege.....	31
Subscription Notifications .....	31
Getting Subscriptions for an Object.....	32
Modifying the Subscriptions for an Object.....	32
<b>Working with Tables .....</b>	<b>33</b>
About Tables .....	33
Operations Supported on Tables .....	33
Loading a Table.....	35
Special Handling in the loadTable Operation .....	36
Working with the Readonly Tables.....	37
Retrieving the Metadata of a Table .....	37
Adding Rows to a Table .....	37
Special Handling in the addRows Operation .....	38

---

Adding a Site to the Sites Tab of an Item .....	38
Adding Suppliers to the Suppliers Tab of an Item .....	38
Adding Suppliers to a Manufacturer Part .....	39
Adding Manufacturer Part to AML of an Item .....	40
Adding Manufacturer Part to the Relationships Tab .....	40
Adding Affected Item to a Change .....	41
Adding Site Specific Item to the BOM Tab .....	41
Adding Site Specific AML to the Manufacturers Tab .....	42
Adding Material to Item table .....	43
Updating Rows in a Table .....	43
Removing Rows from a Table .....	44
Clearing a Table .....	44
Copying Tables .....	45
Redlining a Table .....	45
<b>Working with File Folders and Attachments .....</b>	<b>47</b>
Agile File Folders .....	47
Managing File Folders .....	47
Creating a File Folder .....	47
Checking Out a File Folder .....	48
Setting the Version of File Folder Files .....	49
Cancelling a File Folder Checkout .....	50
Checking In a File Folder .....	50
Deleting the File Folders .....	51
Getting a File from a File Folder .....	51
Getting a File from a File Folder using a Download URL .....	52
Getting a File from a particular Version of File Folder .....	52
Adding Files to a File Folder Object .....	53
Adding Files in a File Folder .....	53
Managing Attachments .....	54
Getting Attachments of an Object .....	54
Getting a Specific Attachment and a File Folder .....	55
Getting a Specific Attachment using a URL .....	55
Adding Attachments to an Object .....	56
Adding attachments by File Reference .....	57
Adding Multiple Attachments into Single Folder .....	57
Adding Files using SOAP Attachment .....	58
Checking Out the Attachments .....	58
Checking Out All the Attachments .....	59
Checking Out Multiple Attachments from a Folder .....	59
Checking In the Attachments .....	59
Checking In an Attachment with FileId Identification .....	60
Deleting the Attachments .....	60

<b>Managing Workflows .....</b>	<b>61</b>
About Agile PLM Workflows .....	61
How the Status of a Change Affects Workflow Functionality .....	61
Getting the Status of a Workflow .....	62
Getting the Workflow of a Routable Object .....	62
Setting a Workflow .....	62
Checking User Privileges .....	63
Adding and Removing Approvers .....	63
Getting Approvers .....	64
Approving a Routable Object .....	64
Rejecting a Routable Object .....	65
Commenting a Change .....	66
Auditing a Change .....	66
Changing the Workflow Status of an Object .....	66
<b>Creating and Loading Queries.....</b>	<b>69</b>
Creating a Query .....	69
Creating a New Query from an Existing Query Object.....	69
Loading a Query .....	70
Executing a Saved Query .....	70
Updating a Saved Query .....	70
Deleting a Query .....	71
Working with Searches .....	71
Agile PLM Searches .....	71
Specifying Search Criteria .....	71
Search Conditions .....	72
Search Operation Keywords .....	72
Specifying Search Attributes .....	73
Getting the Searchable Attributes .....	74
Using Relational Operators .....	74
Using Logical Operators .....	77
Using Wildcard Characters with the Like Operator .....	78
Using Parentheses in Search Criteria .....	78
Using SQL Syntax to specify Search Criteria .....	79
Using SQL Wildcards .....	80
Setting Result Attributes for a Search .....	80
Specifying Result Attributes .....	85
Examples of Searches .....	86
Quick Search .....	86
Advanced Search .....	86
Getting the Searchable Attributes .....	86

<b>Working with Folders.....</b>	<b>89</b>
Loading a Folder.....	90
Creating a Folder.....	91
Working with Folder Elements.....	91
Adding Folder Elements .....	91
Removing Folder Elements .....	92
Getting Folder Elements.....	92
Deleting a Folder .....	92
Renaming a Folder .....	93
<b>Creating and Managing Projects.....</b>	<b>95</b>
About Projects and Projects Objects.....	95
Differences in the Behavior of Projects Objects.....	95
Working with Project Baselines .....	96
Substituting Project Resources .....	96
Delegating Ownership of a Project to Another User .....	96
Locking and Unlocking Projects .....	97
Checking Project's Lock Status .....	97
<b>Publishing Documents .....</b>	<b>99</b>
Document Publishing Web Services Operations .....	99
Loading XML Schema of Objects.....	100
Loading XML Data of Objects .....	100
<b>Performing Administrative Tasks.....</b>	<b>101</b>
Managing Users .....	101
Handling User Passwords .....	101
Changing Login Password.....	101
Changing Approval Password .....	101
Managing Transfer Authorities .....	101
Creating Transfer Authority .....	101
Getting Transfer Authority .....	102
Removing Transfer Authority.....	102
<b>Working with Recipe &amp; Material Workspace .....</b>	<b>103</b>
Performing Searches.....	103
Using Operators .....	103
Searching with Sorting.....	104
Searching with multiple criteria values .....	105
Editing User Data .....	105
Processing Business Objects.....	107

Using cfmXML Schema .....	109
Sample cfmXML message 1 .....	110
LoginInfo .....	110
SessionInfo .....	111
PageInfo .....	111
Response.....	112
Sample cfmXML message 2 .....	112
SearchCriteria .....	113
Attribute .....	114
Relationship .....	114
ObjectKey .....	115
SearchInfo .....	115
Sample cfmXML message 3 .....	115
Payload.....	116
Object .....	116
AdditionalInfo .....	117
<b>Core Operations .....</b>	<b>119</b>
Admin and Metadata Web Services .....	121
getAllClasses .....	121
getSubClasses .....	123
getNode .....	125
getLists .....	127
getAttributes .....	128
getTableMetadata .....	130
getAutoNumbers.....	132
getUsers .....	134
getUserGroups .....	136
convertCurrency .....	137
changeLoginPassword .....	138
changeApprovalPassword.....	139
createTransferAuthority .....	140
getTransferAuthority .....	141
modifyTransferAuthority .....	142
removeTransferAuthority .....	143
getAgileClass .....	144
Attachment Web Services .....	147
getFileAttachment .....	147
addFileAttachment .....	149
checkOutAttachment .....	150
checkInAttachment .....	151
getFileFF .....	152



---

addFileFF .....	153
checkOutFF .....	154
checkInFF .....	155
cancelCheckOutFF .....	156
<b>Business Web Services.....</b>	<b>159</b>
createObject .....	159
getObject .....	162
updateObject .....	164
deleteObject .....	165
undeleteObject .....	166
isDeletedObject .....	167
sendObject .....	168
saveAsObject .....	169
checkPrivilege .....	170
getSubscriptions .....	171
modifySubscriptions .....	173
<b>Collaboration Web Services .....</b>	<b>177</b>
getWorkflows .....	177
getStatus .....	179
auditRObjct.....	181
getApprovers .....	182
changeStatus .....	183
approveRObjct .....	184
rejectRObjct.....	185
setWorkFlow.....	186
addApprovers .....	187
removeApprovers .....	188
commentRObjct.....	189
getReviewers.....	190
<b>Doc Publishing Web Services .....</b>	<b>193</b>
loadXMLSchema .....	193
loadXMLData .....	194
<b>Folder Web Services .....</b>	<b>197</b>
getFolder .....	197
createFolder .....	199
renameFolder .....	200
deleteFolder .....	201
addChildNode.....	202
getChildNode.....	203

removeChildNode.....	204
PC Web Services .....	207
setIncorporate .....	207
getRevisions .....	208
undoRedline .....	210
isRedlineModified .....	211
isIncorporated.....	212
Project Web Services .....	215
createBaseline.....	215
removeBaseline.....	217
getBaselines.....	218
delegateOwnership.....	219
substituteResource.....	220
isLocked .....	221
setLock .....	222
reschedule .....	223
assignUsersFromPool .....	224
Recipe & Material Workspace Web Services.....	227
search.....	227
edit.....	229
processBO .....	231
Reports Web Services.....	233
loadReportSchema.....	233
loadReportData .....	234
Search Web Services .....	237
quickSearch.....	237
advancedSearch .....	239
getSearchableAttributes .....	241
getSearchableClasses.....	243
createQuery.....	244
loadQuery .....	246
executeSavedQuery .....	248
saveAsQuery .....	249
updateQuery.....	251
deleteQuery .....	253
Tables Web Services .....	255
isReadOnlyTable .....	255
clearTable.....	257
copyTable .....	258

---

addRows .....	259
updateRows .....	260
removeRows.....	261
loadTable.....	262
isFlagSet .....	264
User Profile Web Services .....	267
getNotifications .....	267
deleteNotifications .....	276
getMyWorkflowRoutings.....	277
<b>Appendix .....</b>	<b>281</b>
Working with Java Samples .....	281
Building Stubs and Compiling the Samples .....	281
Executing the Samples using ant Task .....	281
Executing the Samples using a Java IDE.....	282
Understanding the Code .....	282
AddFileSOAPAttachment Method .....	283
Helper Methods .....	285
getRowId Method .....	285
getFileId Method.....	287
Troubleshooting.....	291

# Preface

Oracle's Agile PLM documentation set includes Adobe® Acrobat PDF files. The [Oracle Technology Network \(OTN\) Web site](http://www.oracle.com/technetwork/documentation/agile-085940.html) <http://www.oracle.com/technetwork/documentation/agile-085940.html> contains the latest versions of the Agile PLM PDF files. You can view or download these manuals from the Web site, or you can ask your Agile administrator if there is an Agile PLM Documentation folder available on your network from which you can access the Agile PLM documentation (PDF) files.

---

**Note** To read the PDF files, you must use the free Adobe Acrobat Reader version 9.0 or later. This program can be downloaded from the [Adobe Web site](http://www.adobe.com) <http://www.adobe.com>.

---

The [Oracle Technology Network \(OTN\) Web site](http://www.oracle.com/technetwork/documentation/agile-085940.html) <http://www.oracle.com/technetwork/documentation/agile-085940.html> can be accessed through **Help > Manuals** in both Agile Web Client and Agile Java Client. If you need additional assistance or information, please contact My Oracle Support (<https://support.oracle.com>) for assistance.

---

**Note** Before calling Oracle Support about a problem with an Agile PLM manual, please have the full part number, which is located on the title page.

---

## TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

## Readme

Any last-minute information about Agile PLM can be found in the Readme file on the [Oracle Technology Network \(OTN\) Web site](http://www.oracle.com/technetwork/documentation/agile-085940.html) <http://www.oracle.com/technetwork/documentation/agile-085940.html>

## Agile Training Aids

Go to the [Oracle University Web page](http://www.oracle.com/education/chooser/selectcountry_new.html) [http://www.oracle.com/education/chooser/selectcountry\\_new.html](http://www.oracle.com/education/chooser/selectcountry_new.html) for more information on Agile Training offerings.

## Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.





# Introduction to Agile PLM Web Services

**This chapter includes the following:**

---

▪ About Service Oriented Architecture (SOA) .....	1
▪ About Web Services .....	1
▪ About Agile PLM Web Services .....	4
▪ Casual User Interface Integration Examples .....	6
▪ CAD Integration through EC Services .....	9
▪ Building Casual User Interfaces .....	10

## About Service Oriented Architecture (SOA)

Service Oriented Architecture (SOA) is a business-centric IT architecture for building enterprise applications through adaptable and re-usable business processes and services. Each service implements one action such as creating a product record, viewing a BOM table, or updating the Price and Compliance data.

Leading companies are gaining operational efficiencies and business agility through adaptable, re-usable business processes and services built on truly flexible Service-Oriented Architecture (SOA) platforms.

The guiding principles of SOA are:

- Self contained and loosely coupled
- Well defined standards-based interfaces
- Right-sized interfaces
- Location independent and interoperable in a standards-based manner
- Implementation agnostic

One SOA implementation is the Web services approach where the basic unit of communication is a message, rather than an operation. This is often referred to as "message-oriented" services. Web services make functional building-blocks that are accessible over standard Internet protocols and independent of platforms and programming languages. SOA is gaining wide customer adoption because of its reliance on standards-based protocols and enabling rapid development of applications using Web Services. SOA and Web services are supported by most major software vendors.

## About Web Services

Web services are technologies for building distributed applications. These services, which can be made available over the Internet, use a standardized XML messaging system and are not tied to specific operating systems or programming languages. Through Web services, companies can

encapsulate existing business processes, publish them as services, search for and subscribe to other services, and exchange information throughout and beyond the enterprise. Web services are based on universally agreed upon specifications for structured data exchange, messaging, discovery of services, interface description, and business process design.

A Web service makes remote procedure calls across the Internet using:

- HTTP/HTTPS or other protocols to transport requests and responses
- Simple Object Access Protocol (SOAP) to communicate request and response information.

The key benefits provided by Web services are:

- **Service-oriented Architecture** – Unlike packaged products, Web services can be delivered as streams of services that allow access from any platform. Components can be isolated; only the business-level services need be exposed.
- **Interoperability** – Web services ensure complete interoperability between systems.
- **Integration** – Web services facilitate flexible integration solutions, particularly if you are connecting applications on different platforms or written in different languages.
- **Modularity** – Web services offer a modular approach to programming. Each business function in an application can be exposed as a separate Web service. Smaller modules reduce errors and result in more reusable components.
- **Accessibility** – Business services can be completely decentralized. They can be distributed over the Internet and accessed by a wide variety of communications devices.
- **Efficiency** – Web services constructed from applications meant for internal use can be used for externally without changing code. Incremental development using Web services is relatively simple because Web services are declared and implemented in a human readable format.

## Core Technologies

*Oracle's Agile Web services use industry standard core technologies.* The bulleted list that follows lists these technologies. Each core technology is explained in detail in the topics that follow.

- Web Services Description Language (WSDL)
- XML and XML Schema
- Simple Object Access Protocol (SOAP)

## Web Services Description Language (WSDL)

WSDL is an XML-based format for describing the interface of a Web service. WSDL describes the endpoints, location, protocol binding, operations, parameters, and data types of all aspects of a Web service:

- The WSDL that describes a Web service has the following characteristics:
  - It is published by the service provider.
  - It is used by the client to format requests and interpret responses.
  - It can be optionally submitted to a registry or service broker to advertise a service.
- Additionally, WSDL describes the following:



- The operations that are provided by a Web service.
- The input and output message structures for each Web service operation.
- The mechanism to contact the Web service.

## XML and XML Schema

A WSDL file is published as an XML file. Document/Literal is required as part of the WS-I interoperability standard. This standard sets the basis for modern Web service usage.

- **Document** – The payload for an operation, however complex, must be defined in a single XML element.
- **Literal** – The definition of single XML element must be described by an XML Schema embedded in the WSDL file.

When using Document/Literal formatting, the WSDL file will contain an XML Schema definition that defines all messages and data types that are used for a particular service. The payload itself will consist entirely of XML data structures.

## Simple Object Access Protocol (SOAP)

SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. SOAP uses XML to define an extensible messaging framework.

SOAP messages consist of the following:

- An envelope for wrapping messages, including addressing and security information.
- A set of serialized rules for encoding data types in XML.
- Conventions for a procedure call and, or response.

## Web Services Architecture

You can view Web services architecture in terms of roles and the protocol stack:

- Web services roles:
  - **Service provider** – This provides the service by implementing it and making it available on the Internet.
  - **Service requester** – This is the user of the service who accesses the service by opening a network connection and sending an XML request.
  - **Service registry** – This is a centralized directory of services where developers can publish new services or find existing ones.
- Web services protocol stack:
  - **Service transport layer** – This layer uses the HTTP protocol to transport messages between applications.
  - **XML messaging layer** – This layer encodes messages in XML format using SOAP to exchange information between computers. It defines an envelope specification for encapsulated data that is transferred, the data encoding rules, and remote procedure call (RPC) conventions.

- **Service description layer** – This layer describes the public interface to a specific Web service using the Web Service Description Language (WSDL) protocol. With WSDL, it defines an XML grammar to describe network services. The operations and messages are described abstractly, and then bound to a network protocol and message format. WSDL allows description of endpoints and their messages regardless of what message formats or network protocols are used to communicate.
- **Service discovery layer** – This layer centralizes services into a common registry using the Universal Description, Discovery, and Integration (UDDI) protocol. UDDI is a platform-independent, XML-based registry for businesses worldwide to list themselves on the Internet.

## About Agile PLM Web Services

Implementation of Agile PLM Web Services adheres to the following principles:

- Well defined standards based discoverable Interface
- XML based Web Service Framework - Apache Axis 1.4
- Modularized PLM Schema (XSD) and WSDL for easy maintenance
- Standards-based WSDL to ensure compatibility across various clients (.NET, Java, and BPEL)
- Batch APIs wherever applicable for better performance
- Web Service versioning for backward compatibility

Agile PLM Web Services expose all key PLM functionalities in the following services.

- **Agile PLM Core Web Services** – These services support functionalities provided by PLM solutions such as PC, PQM, PCM, PPM, PG&C. See [Agile PLM Core Web Services Operations](#) on page 119.
- **Agile PLM EC Web Services** – These services support functionalities provided by Agile PLM's Engineering Services (EC) solution. See Agile PLM EC Services Guide at [Oracle Technology Network \(OTN\) Web site](http://www.oracle.com/technetwork/documentation/agile-085940.html) <http://www.oracle.com/technetwork/documentation/agile-085940.html>.

## Agile PLM Core Web Services

Agile PLM Core Web Services are a set of services for the following PLM functionalities:

- Business Object CRUD (Create, Read, Update, Delete) data services
- Collaboration services
- Meta data Services
- Search Services
- Attachment Services
- Table Services

## Agile PLM EC Services

Agile PLM Engineering Collaboration (EC) Services are a set of Business Services that supplement PLM's Core Web Services for CAD use cases. They also offer a set of higher level BPEL orchestration services. Customers and partners can build next generation MCAD and ECAD connectors utilizing Agile PLM Web Services and Engineering Collaboration Services.

Some of the benefits are:

- Significantly Improves WAN performance for CAD connectors because the bulk of the logic is deployed to the server
- Makes it easier for development partners and customers to implement CAD connectors
- Provides the unique and interface friendly API Name field to access PLM metadata

## Agile Recipe & Material Workspace Web Services

The Recipe & Material Workspace module in Agile application caters to the needs product lifecycle management of the pharmaceutical development industry. It is made up of several dimensions such as Recipe (Instructions), Equipment, Material, Analytical (Test and Assays), Environment, Standards and People. These dimensions enable drug manufacturers to conduct the preparation, execution and analysis necessary during the scale-up life cycle of a substance across multiple pilot plants, located in disparate geographic locations. It also helps scale up material production in a systematic and reproducible manner.

The RMW Web Services are a set of Business Services that supplement Agile PLM's Core Web Services for Pharmaceutical industry use cases. They also offer a set of higher level BPEL orchestration services. Customers and partners can build next generation applications to carry out business object processing, searches and data editing.

## Agile PLM Web Service Authentication and Performance

In implementations where scalability is critical, a lightweight context management facility for authentication is available and its use is recommended. With this facility, authentication is managed using a combination of user credentials and a `sessionID` token:

- When user credentials are presented in the SOAP header of a Web service request, formal authentication is performed prior to the application execution of the Web service operation. If the authentication succeeds, the operation proceeds and a special SessionID token are placed in the SOAP header of the Web service reply.
- Whenever the `sessionID` is included by the client in subsequent Web service requests, that `sessionID` will be used to restore cached session information, thus bypassing the substantially more expensive process of re-executing the authentication. Note that, when presented with both the `sessionID` and a valid set of user credentials, an attempt will be made to use the `sessionID` before resorting to the user credentials and re-authentication. As expected, the session that is being tracked by the `sessionID` is subject to expiration and other security checks.

The facility is a distinct alternative to the basic authentication standard described by WS-Security. Using the `UserName` token as provided in WS-Security, while fully supported as part of Agile PLM's WSI Basic Profile compliance, will not yield the same benefit as using the higher-performance

session optimization facility provided by the Agile PLM implementation.

## Impact on Existing Agile PLM Extensions and Services

Agile PLM provides tools and process extensions to customize the Agile PLM to meet unique user requirements, provide access to external databases, extend automation capabilities, and develop UI extensions. These tools and services are listed below. Agile PLM Web Services implementation has no impact on these capabilities; they are in addition to the existing services.

- **Agile SDK** – The SDK is a set of Java APIs that enable building custom applications to access or extend the Agile PLM server functionalities. **For information, refer to *Agile PLM SDK Developer Guide*.**
- **Agile Integration Services (AIS)** – AIS is a collection of predefined Web Services in the Agile Integration framework that enable communication between PLM server and disparate database. **For information, refer to *Agile PLM AIS Developer Guide*.**
- **Agile Content Services (ACS)** – ACS is a process for transferring data to other Agile PLM solutions or to any other external system. **For information, refer to *Agile PLM ACS User Guide*.**
- **Process Extensions (PX)** – PX is a framework for extending the functionality of the Agile PLM system. The functionality can be server-side extensions such as custom automations, or client-side functionality such as new commands added to the Java/Web Client's Actions or Tools menus. **For information, refer to *Agile PLM SDK Developer Guide*.**
- **Web Service Extensions (WSX)** – WSX is a Web service engine that enables communication between Agile PLM and internal and external systems. **For information, refer to *Agile PLM SDK Developer Guide*.**
- **Dashboard Management Extensions (DX)** – Similar to PX, DX extends the functionalities of the Agile PLM system. **For information, refer to *Agile PLM SDK Developer Guide*.**

## Casual User Interface Integration Examples

Agile Web Client and Agile Java Client are targeted towards those who use the more complex product lifecycle management features of Agile PLM on a daily basis to perform assigned tasks and duties. There is also another set of users who use the auxiliary capabilities of Agile PLM to perform lightweight tasks such as document management, importing compliance and price data, or approving ECO and Sales RFQ.

The tools of choice for these users are the popular desktop products provided by Microsoft Office or Adobe Acrobat, Mobile devices. They prefer simple user-friendly interfaces, for example:

- Microsoft Word and Acrobat for document management
- Microsoft Excel to import price and compliance data
- Oracle WebCenter and Oracle Application Development Framework (ADF) for simple document management tasks
- Oracle WebCenter and ADF for simple item management tasks
- Mobile devices to access sales RFQ
- Mobile devices to access ECO Approval

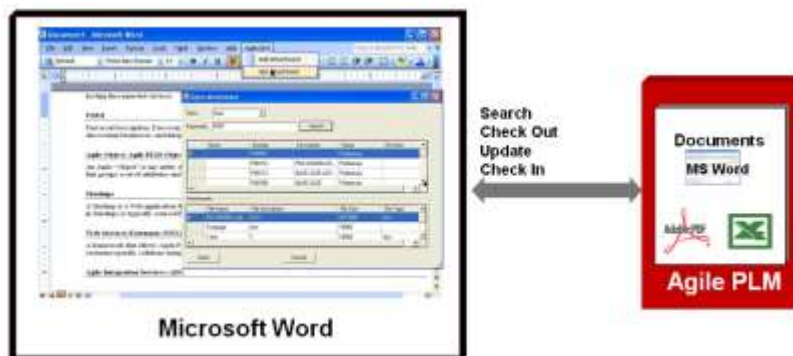
- Microsoft Sharepoint for simple document management tasks

## User Interface Integration - MS Word

This example demonstrates document management capabilities of PLM's Web Services. Currently, when casual users want to view or update a document in Agile PLM, they do so by logging in to the Web Client to retrieve and view the Word document. The steps are:

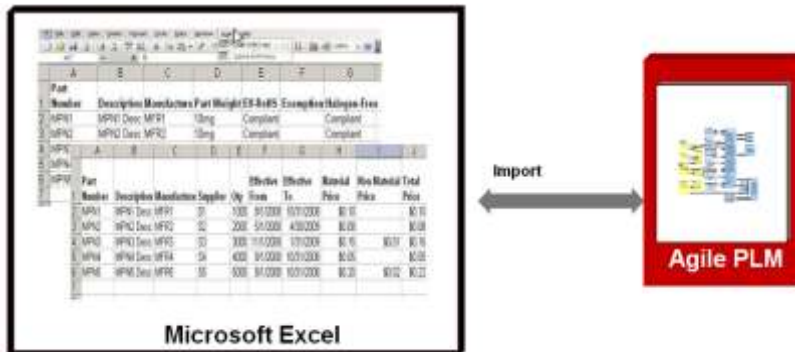
1. Log in to PLM Client
2. Search and locate the document
3. Check out the document (in Word)
4. Modify the documents (in Word)
5. Check In the document
6. Log out

Using Agile PLM's Web Services, the casual user directly accesses Agile PLM documents from MS Word. This simple UI will encourage and accelerate greater user participation. Agile PLM is transparent to this class of users which eliminates training and exposure in PLM Web Client.



## User Interface Integration - MS Excel

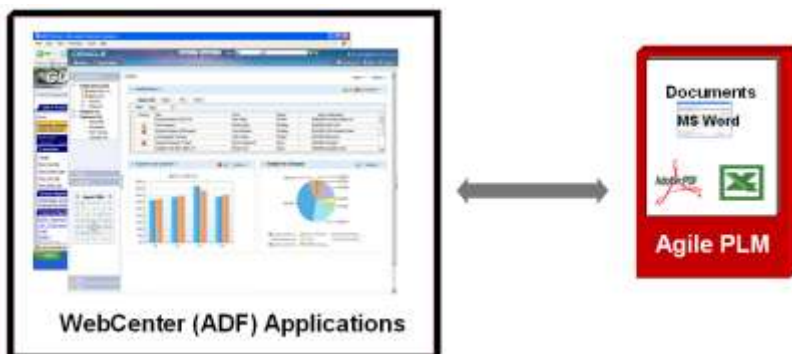
This is similar to MS - Word integration. In this case, the casual user is one of your partners and suppliers. Using PLM's Web Services, you can provide a simple UI in Excel template for suppliers and partners. Then when necessary, suppliers import information such as compliance and price data directly into PLM system from Excel. Benefits include greater and more convenient supplier participation in the PLM process with no training in Agile PLM Web Client.



## User Interface Integration - Portals and Agile Web Client

Before PLM Web Services, the practice was to create custom Web applications using Agile PLM SDK with various tools and technologies. With Web Services, you can build rich Web applications in Oracle Web Center (and ADF) by taking advantage of Web 2.0 UI and mobile services.

Once you develop the custom UI Web application for casual users, you can also integrate the custom UI with Agile Web Client using Agile PLM's URL Process Extensions (refer to *Agile PLM SDK Developer Guide*) and Smart URL features.



## User Interface Integration - Mobile ADF

One of the key demands in Agile PLM installations is mobile access for management and executive personnel. One such example is ECO Approval by the senior or management staff using mobile devices. PLM's Web Services enable developing simple ECO Approval applications for users of mobile devices.

The following illustrations depict a sales RFQ implementation from a sales manager's perspective:

Using the mobile device's browser, the sales manager launches the Mobile application built using Agile Web Services. The first screen is the Search RFQ screen. The second is the RFQ Details screen and the third, the Send RFQ screen.



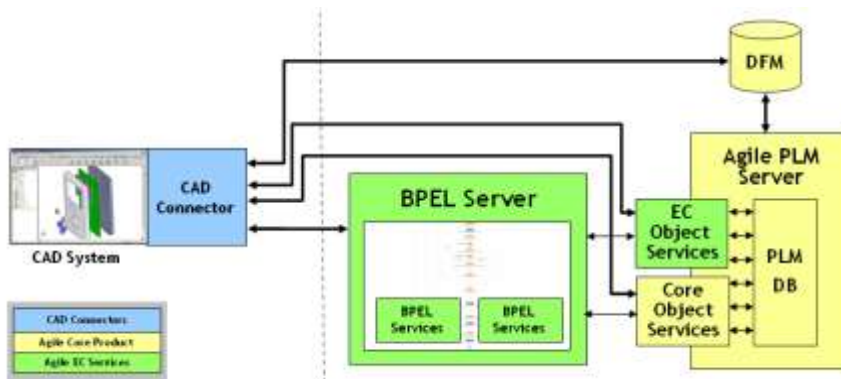
## CAD Integration through EC Services

Customers and partners can build next generation MCAD and ECAD connectors with the aid of Agile PLM Core and Engineering Collaboration Web Services. The benefits were summarized in [Agile PLM EC Services](#) on page 5.

Some of the EC services are orchestrated using Business Process Execution Language (BPEL). BPEL, short for Web Services Business Process Execution Language is an executable language for specifying interactions with Web Services. Processes in Business Process Execution Language export and import information by using Web Service interfaces exclusively.

The CAD integration and the role of BPEL server is shown in the following illustration.

**Figure 1: CAD integration architecture**



## Building Casual User Interfaces

The following paragraphs describe the tools and the steps in developing some the UI integration examples in MS Office and Oracle Web Center (and ADF) environments.

### Developing User Interfaces for MS Office

Microsoft supports building UI integration interfaces by providing the *Microsoft Office Add-in* (a piece of code) for this purpose. MS Office Add-in support integration at Application level and document level.

To develop the MS Word Add-in with PLM, the following tools and applications are necessary:

- Application software
  - Microsoft Visual Studio 2005/2008
  - Dot NET framework 3.5
  - Agile PLM (v9.3 or above) server
  - Microsoft Word 2003/2007
- Programming languages
  - C#
  - Visual Basic for .NET
  - Microsoft Visual C++/ATL
- Plug-in templates
  - Shared Add-in Extensibility template

---

**Note** The *Shared Add-in Extensibility* templates are used to deploy a single add-in onto multiple Microsoft Office applications (common add-ins across Word, Excel, and other office applications). This Add-in is always installed only at the application-level.

---

- Office 2003/2007 Add-in template

#### Steps in developing an MS Office Add-in:

1. Evaluate Add-in type: application level versus document level
2. Evaluate programming language: C#, Visual Basic
3. Create a project in Microsoft Visual Studio 2005/2008
4. Generate the C#/Visual Basic Stubs from Agile WSDL  
For information on Stubs, see [Generating and Initializing the Stubs](#) on page 15.
5. Create Windows Forms
6. Bind data to UI controls
  - Populate documents with data from Agile Web Services



7. Build & test the Add-in
8. Deploy the Add-in
9. Extend Agile PLM (v9.3 or above) Samples to fit your business needs:
  - MS Word Document Management - Application Level Add-In
  - MS Excel – Import BOM/Price/ Compliance - document Level Add-in

**Note** The source code for these two MS Office Add-ins is available for download from Oracle OTN Website. A good source of information for developing an MS Office Add-in is the MSDN forum.

## Developing User Interfaces for Oracle WebCenter and ADF

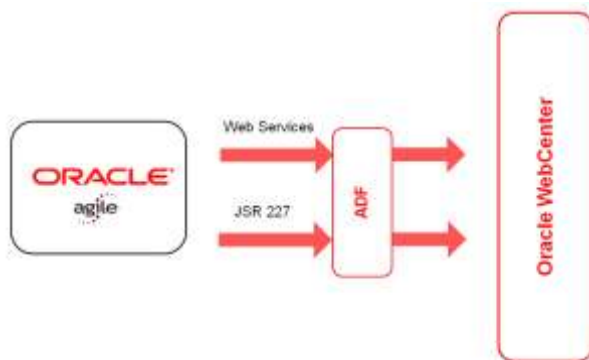
This section provides basic information to develop the following UIs in Oracle Web Center and ADF environments.

- Document Management UI in Oracle WebCenter (and ADF 10g)
- Item Management UI in Oracle WebCenter (and ADF 11g)
- Sales RFQ UI in Mobile device
- ECO Approval UI in Mobile device

You need the following tools and applications to develop the Oracle WebCenter (and ADF) with PLM:

- Software
  - Oracle jDeveloper 10g/11g
  - Agile PLM (v9.3 or above) server
- Programming Languages
  - Java

**Figure 2: Oracle WebCenter with Agile PLM**



### Steps in developing ADF applications

1. Create a Project in jDeveloper
2. Generate the Java Stubs from Agile WSDL

3. Map XML schema to Java Classes
4. Create UI forms
5. Create page flow
6. Bind data to UI controls
7. Build and test the applications
8. Deploy the applications
9. Extend Agile (v9.3 or above) samples to meet business needs:
  - Document management
  - Item management
  - Sales RFQ

For information on Oracle Web Center, ADF, and jDeveloper, visit Oracle Web site at:  
<http://www.oracle.com/technology/products/webcenter/index.html> and  
<http://www.oracle.com/technology/products/adf/index.html>

# Getting Started with Agile Web Services

**This chapter includes the following:**

---

▪ Operational Environment .....	14
▪ Generating and Initializing the Stubs .....	15
▪ Understanding the MessageElement .....	15
▪ Agile Attributes without API Names .....	18
▪ Understanding the Web Services Responses .....	19

## Before Building a Web Services Client

Verify that the following are in place:

- The following jars are present in classpath:

- axis.jar
- axis-ant.jar
- commons-discovery.jar

- Ant libraries are present.

- The Ant build file contains the following:

```
<path id="build.classpath">
  <fileset dir="${axislib.dir}">
    <include name="**/*.jar" />
  </fileset>
</path>
<taskdef resource="axis-tasks.properties">
  <classpath refid="build.classpath"/>
</taskdef>
<target name="wsdl2java-Generate-Client">
  <echo message="Generating all the client side stubs"/>
  <axis-wsdl2java
    all="true"
    output="./src"
    verbose="true"
    url="http://<host>:<port>/core/services/<serviceName>?wsdl">
  </axis-wsdl2java>
</target>
```

Replace the `<serviceName>` with the name of the Web Service. For example, `BusinessObject`, `Collaboration` and so on.

## Operational Environment

Development platforms vary in their SOAP implementations. Implementation differences in certain development platforms may prevent access to some or all of the features in the API. If you are using Visual Studio for .NET development, it is recommended that you use Visual Studio 2003 or higher.

Agile PLM Application	Release 9.3.1
Default Web Services Engine	Apache Axis 1.4
Java 2 Platform Standard Edition Development Kit	5.0

## Standards Compliance

The Agile PLM Web Services are implemented in compliance with the following standards:

Standard	Location
Simple Object Access Protocol (SOAP) 1.1/1.2	<a href="http://www.w3.org/TR/2000/NOTE-SOAP-20000508/">http://www.w3.org/TR/2000/NOTE-SOAP-20000508/</a>
Web Service Description Language (WSDL) 1.2	<a href="http://www.w3.org/TR/2001/NOTE-wsdl-20010315">http://www.w3.org/TR/2001/NOTE-wsdl-20010315</a>
WS-I Basic Profile 1.1	<a href="http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html">http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html</a>
XML Schema 1.1	<a href="http://www.w3.org/XML/Schema">http://www.w3.org/XML/Schema</a>

## Web Services Engines

All Application Server vendors, such as Oracle, BEA, IBM, have built-in Web Services infrastructure solutions that are integrated with their application servers. For non-web services integrated applications, there are stand-alone products, such as AXIS from Apache, which provide Web Services infrastructure that can be integrated with different application servers.

The following Web Services Engines are supported:

- Oracle Apps Server Web Service Infrastructure
- WebLogic Web Service Infrastructure
- Axis - version 1.4
- Axis2 - version 1.4

<b>Important</b> Axis 1.4 is the default Web Services Engine for Agile PLM Release 9.3.1
--

## Generating and Initializing the Stubs

The Stub acts as a gateway for client side objects and all outgoing requests to server side objects that are routed through it. The stub wraps client object functionality and by adding the network logic ensures the reliable communication channel between client and server. The stub can be written up manually or generated automatically depending on chosen communication protocol.

For Agile Web Services to function successfully, you first need to create Agile PLM Server Stubs, and initialize the Client side Stubs.

### Generating Agile Stubs

Execute the Ant target `wsdl2java-Generate-Client`.

All the stubs are created in `src` folder.

### Initializing the Client Stubs

In the following sample, the generated stubs are being initialized for Business Object Web Services client. You may adapt it for other Web Services.

```
String SERVER_URL = "http://<host>:<port>/core/services/BusinessObject";
String USERNAME = "admin";
String PASSWORD = "agile";
BusinessObjectServiceLocator locator = new BusinessObjectServiceLocator();
BusinessObject_BindingStub businessObjectStub =
    (BusinessObject_BindingStub) locator.getBusinessObject(new java.net.URL(SERVER_URL));
((org.apache.axis.client.Stub) businessObjectStub).setUsername(USERNAME);
((org.apache.axis.client.Stub) businessObjectStub).setPassword(PASSWORD);
```

## Understanding the MessageElement

- A MessageElement is a part of a Request that specifies attributes of Agile Objects, which can use Agile API Names.
- Most MessageElements are String Type, while some can be Unit of Measure Type or AgileListEntryType.
- A MessageElement can be assigned any Tag Name. When API name is used as Tag Name, you need not pass the 'attributeld'.

```
MessageElement dataCell = new MessageElement(namespaceUri, "Numeric01");
dataCell.setObjectValue(9144.0);
```

- When API name is not used as Tag Name, attributeld has to be explicitly passed as an XML attribute having the name 'attributeld'

```
MessageElement dataCell = new MessageElement(namespaceUri, "key");
dataCell.setObjectValue(9144.0);
dataCell.addAttribute(namespaceUri, SchemaConstants.attributeId.getValue(),
    "Attribute ID");
```

## Obtaining the API Names and Attribute IDs

To obtain the API Names, open the desired Class in Agile Java Client. You will find all the API Names in the General Information page or under the General Information tab.

## Special Handling of MessageElements

The following MessageElement Types require special handling, as described below.

### Unit of Measure

The attributes of an Agile object that require 'Unit of Measure' as an input type are updated with the UOM values. These values are denoted by the Unit of Measure object. The corresponding object is **AgileUnitOfMeasureType**.

To do this, you have to send the data as an instance of AgileUnitOfMeasureType. In addition, you need to pass the corresponding Namespace URI as an attribute.

The format is:

```
messageElement.addAttribute(PREFIX, NAMESPACEURI, "type", CLASSNAME);
```

You can choose any meaningful value for PREFIX. The type should be passed as "type".

You are required to send correct values for NAMESPACEURI and CLASSNAME. For example:

```
COMMONNAMESPACEURI = "http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
CLASSNAME = "AgileUnitOfMeasureType"
```

You can define your own namespaceUri or use COMMONNAMESPACEURI.

#### Example: MessageElement for a UOM

```
AgileUnitOfMeasureType uom = new AgileUnitOfMeasureType();
uom.setUnitName("Kilogram");
uom.setUnitValue(1000.0);
MessageElement dataCell=new MessageElement(namespaceUri, "mass");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, SchemaConstants.type.getValue(),
"AgileUnitOfMeasureType");
dataCell.setObjectValue(uom);
message[0] = dataCell;
```

## Multilist and List

The corresponding object is **AgileListEntryType**. You are required to pass the namespace attribute.

#### Example: MessageElement for a Multilist

```
AgileListEntryType multilist01 = new AgileListEntryType();
SelectionType[] multiSelect = new SelectionType[3];
multiSelect[0] = new SelectionType();
multiSelect[0].setValue("Canceled");
multiSelect[1] = new SelectionType();
multiSelect[1].setValue("Complete");
multiSelect[2] = new SelectionType();
multiSelect[2].setValue("Accepted");
multilist01.setSelection(multiSelect);
```

```
MessageElement dataCell = new MessageElement(namespaceUri, "multilist01");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, SchemaConstants.type.getValue(),
    "AgileListEntryType");
dataCell.setObjectValue(multilist01);
message[1] = dataCell;
```

### Example: MessageElement for a List

```
AgileListEntryType list01 = new AgileListEntryType();
SelectionType[] selection = new SelectionType[1];
selection[0] = new SelectionType();
selection[0].setValue("Alternate");
list01.setSelection(selection);
MessageElement dataCell = new MessageElement(namespaceUri, "list01");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, SchemaConstants.type.getValue(),
    "AgileListEntryType");
dataCell.setObjectValue(list01);
message[1] = dataCell;
```

### List of Objects

In certain cases, Agile SDK expects a list of `IDataObject` to be passed. For such cases, Agile Web Services use `AgileObjectListEntryType`.

```
AgileObjectListEntryType multilist01 = new AgileObjectListEntryType();
ObjectReferentIdType[] obj = new ObjectReferentIdType[1];
obj[0] = new ObjectReferentIdType();
obj[0].setClassIdentifier("8750");
obj[0].setObjectIdentifier("SAP0265");
multilist01.setSelection(obj);
MessageElement dataCell = new MessageElement(namespaceUri, "supplier");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, SchemaConstants.type.getValue(),
    "AgileObjectListEntryType");
dataCell.setObjectValue(multilist01);
message[0] = dataCell;
```

### Money

The corresponding object is `AgileMoneyType`. You are required to pass the *namespace* attribute.

```
AgileMoneyType money = new AgileMoneyType();
money.setAmount(997777.9);
money.setCurrency("USD");
MessageElement dataCell = new MessageElement(namespaceUri, "money01");
dataCell.setObjectValue(money);
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, SchemaConstants.type.getValue(),
    "AgileMoneyType");
message[0] = dataCell;
```

## Date

Java provides an object either as Date or as Calendar objects. You are required to pass the *namespace* attribute.

Date is a special case. Even though it is an XSD type, you must pass URI for Date.

### Example: MessageElement for Date

```
xsdnamespace = "http://www.w3.org/2001/XMLSchema"
MessageElement dataCell = new MessageElement(namespaceUri, "date01");
dataCell.addAttribute(XSIPREFIX, xsdnamespace, SchemaConstants.type.getValue(),
    "dateTime");
dataCell.setObjectValue(new Date());
message[0] = dataCell;
```

## User/Supplier/Customer/Analyst

The corresponding object is `ObjectReferentIdType`. You are required to pass the *namespace* attribute.

### Example: MessageElement for a User

```
ObjectReferentIdType user = new ObjectReferentIdType();
user.setObjectIdentifier("EMS1");
user.setClassIdentifier("supplier");
MessageElement dataCell=new MessageElement(namespaceUri, "supplier");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, SchemaConstants.type.getValue(),
    "ObjectReferentIdType");
dataCell.setObjectValue(user);
message[0] = dataCell;
```

### Example: MessageElement for a Customer

```
ObjectReferentIdType customer = new ObjectReferentIdType();
customer.setObjectIdentifier("DEMO CUSTOMER 1");
customer.setClassIdentifier("customer");
MessageElement dataCell=new MessageElement(namespaceUri, "customer");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, SchemaConstants.type.getValue(),
    "ObjectReferentIdType");
dataCell.setObjectValue(customer);
message[0] = dataCell;
```

---

**Note** Other values, like String, numbers etc can be passed as they are. However, numbers should not be passed as strings.

---

## Agile Attributes without API Names

The following attributes do not have an API name. You are required to use the Attribute IDs, listed below. These values have been picked from Agile SDK Constants. For more information on this,



refer to *Agile SDK Developer Guide*.

## Item Constants

TABLE_REDLINEBOM	new Integer(-803);
TABLE_REDLINEMANUFACTURERS	new Integer(-1491);
TABLE_REDLINETITLEBLOCK	new Integer(-801);
TABLE_REDLINEPAGETWO	new Integer(-810);
TABLE_REDLINEPAGETHREE	new Integer(-1501);
FLAG_IS_REDLINE_MODIFIED	new Integer(-101);
FLAG_IS_REDLINE_REMOVED	new Integer(-102);
FLAG_IS_REDLINE_ADDED	new Integer(-103);

## User Constants

ATT_LOGIN_PASSWORD	new Integer(-1);
ATT_APPROVAL_PASSWORD	new Integer(-2);
ATT_SUPPLIER	new Integer(-3);
ATT_LOCALE	new Integer(-4);
ATT_TIMEZONE	new Integer(-5);
ATT_DATEFORMAT	new Integer(-6);
ATT_DATETIMEFORMAT	new Integer(-7);

# Understanding the Web Services Responses

## Response Status Code

The response obtained from every Web Service call contains a response **statusCode**, which indicates the success or failure of a Web Service operation. These Status Codes are of four types:

- **SUCCESS**- indicates that all Web Services in the batch were executed successfully and that all operations worked as intended.
- **FAILURE** - indicates that all Web Services in the batch failed during execution, indicating the intended operations were not performed.
- **WARNING** - indicates that while Web Services in the batch were successfully executed, however certain warnings were also encountered during the execution. These warnings need to be analyzed by the client to verify that all operations worked as intended.
- **PARTIAL\_SUCCESS** - indicates a partial success in the execution of batch Web Services when

one or more but not all batch requests have failed. Even if a single Web Service fails among a batch of Web Services, the response status code will indicate PARTIAL\_SUCCESS.

## Exceptions and Warnings

When an operation is not successful, the system will throw an Exception or a Warning.

- In case of **FAILURE**, an exception is issued, while a warning may or may not be issued.
- In case of **WARNING**, only a warning is issued.

When the status is WARNING, the outcome of the operation is unknown. You are required to check it manually whether the operation was successful or not.

The Exceptions require code correction or system administration, while the Warnings can be resolved as described in [Working with Warnings](#) on page 21.

The response header for Web Services calls consists of a list of exceptions and warnings populated as **AgileExceptionListType** and **AgileWarningListType** objects. The application client must check for exceptions and warnings at all times to ensure that the code has performed all operations as intended.

The exception and warning lists contain a reference element 'id' which may be used to identify the corresponding requested in the batch that was the source of the exception(s) or warning(s).

Refer to the schema for dealing with response objects for a particular Web Service.

---

**Note** The code snippets provided in the first section of this document discuss only about the Request Objects. See [Core Operations Reference](#) on page 119 for complete sample code.

---

### Example: Getting Exceptions and Warnings

```
if ( !approveRObjResponseType.getStatusCode().toString().equals(
ResponseStatusCode.SUCCESS.getValue() ) ){
    AgileExceptionListType[] agileExceptionListType =
approveRObjResponseType.getExceptions();
    if (agileExceptionListType!=null)
        for (int i=0; i<agileExceptionListType.length; i++){
            AgileExceptionType exceptions[] = agileExceptionListType[i].getException();
            for (int j=0; j<exceptions.length; j++){
                System.out.println("Exception Id:" + exceptions[j].getExceptionId() +
"\nMessage: " +
                    exceptions[j].getMessage() );
            }
        }
    AgileWarningListType agileWarningListType[] =
approveRObjResponseType.getWarnings();
    if (agileWarningListType!=null)
        for (int i=0; i<agileWarningListType.length; i++){
            AgileWarningType warnings[] = agileWarningListType[i].getWarning();
            for (int j=0; j<warnings.length; j++){
                System.out.println("Warning Id: " + warnings[j].getWarningId() + "\nMessage:
" +
                    warnings[j].getMessage() );
            }
        }
    }
```

```
}
```

## Working with Warnings

---

**Note** By default, all warnings are enabled.

---

You can work with warnings in following ways:

- Use **setWarningResolution** to selectively Enable or Disable a select set of warnings.

```
AgileWarningResolutionType warningRes[] = new AgileWarningResolutionType[1];
warningRes[0] = new AgileWarningResolutionType();
warningRes[0].setId(182);
warningRes[0].setResolution(AgileWarningResolutionConstantsType.DISABLE);
approveObjectRequestType.setWarningResolution(warningRes);
```

- Use **diasableAllWarnings** function to disable ALL the warnings

```
approveObjectRequestType.setDisableAllWarnings();
```

- Enable a select set of warnings and disable the rest with a combination of **diasableAllWarnings** and **setWarningResolution**.

```
approveObjectRequestType.setDisableAllWarnings();
AgileWarningResolutionType warningRes[] = new AgileWarningResolutionType[1];
warningRes[0] = new AgileWarningResolutionType();
warningRes[0].setId(182);
warningRes[0].setResolution(AgileWarningResolutionConstantsType.ENABLE);
approveObjectRequestType.setWarningResolution(warningRes);
```



# Working with Business Objects

This chapter includes the following:

---

▪ Getting an Object.....	23
▪ Creating an Object.....	24
▪ Saving As a New Object.....	25
▪ Deleting and Undeleting an Object.....	27
▪ Updating an Object.....	28
▪ Getting the Status of an Object.....	28
▪ Getting the AutoNumbers .....	29
▪ Getting all the Classes.....	29
▪ Getting the Subclasses of a Class.....	29
▪ Getting Agile Classes .....	30

This chapter describes how to work with the Agile PLM Business Objects and provides sample code snippets.

## Getting an Object

To get an Agile PLM object, use the `getObject` operation. This operation lets you specify the `objectType` and `objectNumber` parameters.

An `objectType` is the API name or ID of a Subclass. For example, a 'Part' is an `objectType` of the Agile Class 'Item'; ECO is an `objectType` of the Agile Class 'Change'. An `objectNumber` is number of the Agile Object being retrieved.

The actual information about the object is obtained through the response in the form of `AgileObjectType` objects.

Use the following syntax to get an object by specifying `objectType` and `objectNumber` parameters, as shown in examples.

```
GetObjectRequestType getObjectRequestType = new GetObjectRequestType();
AgileGetObjectRequest agileGetObjectRequest[] = new AgileGetObjectRequest[1];
agileGetObjectRequest[0] = new AgileGetObjectRequest();
agileGetObjectRequest[0].setClassIdentifier("objectType");
agileGetObjectRequest[0].setObjectNumber("objectNumber");
```

### Example: Getting an Item

```
GetObjectRequestType getObjectRequestType = new GetObjectRequestType();
AgileGetObjectRequest agileGetObjectRequest[] = new AgileGetObjectRequest[1];
agileGetObjectRequest[0] = new AgileGetObjectRequest();
agileGetObjectRequest[0].setClassIdentifier("Part");
agileGetObjectRequest[0].setObjectNumber("P00001");
```

**Example: Getting a Change**

```
GetObjectRequestType getObjectRequestType = new GetObjectRequestType();
AgileGetObjectRequest agileGetObjectRequest[] = new AgileGetObjectRequest[1];
agileGetObjectRequest[0] = new AgileGetObjectRequest();
agileGetObjectRequest[0].setClassIdentifier("ECO");
agileGetObjectRequest[0].setObjectNumber("ECO-0001");
```

---

**Note** The getObject operation request for the resource types *User, Owner and UserGroup* returns a response in integer values. These values are mapped as: 11605 = User, 18044 = Owner, 11885 = User Group.

---

## Special Handling in the getObject Operation

To get certain Agile Objects, the getObject operation requires an additional parameter to be set. This parameter, `setOptions(propertyType)`, accepts a name-value pair - **propertyName** and **propertyValue**.

**Example: Getting an Manufacturer Part**

```
GetObjectRequestType getObjectRequestType = new GetObjectRequestType();
AgileGetObjectRequest agileGetObjectRequest[] = new AgileGetObjectRequest[1];
agileGetObjectRequest[0] = new AgileGetObjectRequest();
agileGetObjectRequest[0].setClassIdentifier("ManufacturerPart");
agileGetObjectRequest[0].setObjectNumber("manufPartNumber");
PropertyType[] propertyType = new PropertyType[1];
propertyType[0] = new PropertyType();
propertyType[0].setPropertyName(SchemaConstants.manufacturer_name.getValue());
propertyType[0].setPropertyValue(manufName);
agileGetObjectRequest[0].setOptions(propertyType);
```

## Creating an Object

To create a new Agile PLM object, use the createObject operation. This operation requires you to specify the **objectType** parameter, for example, a *Part*.

**Example: Creating a Part**

```
CreateObjectRequestType createObjectRequestType = new CreateObjectRequestType();
AgileCreateObjectRequest agileCreateObjectRequest[] = new AgileCreateObjectRequest[1];
agileCreateObjectRequest[0].setClassIdentifier("Document");
AgileRowType row_1 = new AgileRowType();
MessageElement messages_1[] = new MessageElement[2];
String namespaceUri = null;
messages_1[0] = new MessageElement(namespaceUri, "number");
messages_1[0].addTextNode(documentNumber);
messages_1[1] = new MessageElement(namespaceUri, "description");
messages_1[1].addTextNode("Doc Desc");
```

---

**Note** Agile Web Services do not support setting the Life Cycle Phase (LCP)/workflow status attribute of an object while you are creating that object. This is because the necessary settings for LCP are not available until the object is created.

---

## Saving As a New Object

You can save an existing Agile Object as a new object by using the `saveAsObject` operation. This operation calls `AgileSaveAsObjectRequestType`, which requires the values of `objectName`, `objectNumber` and `newObjectName`.

You can specify a `newObjectNumber` as a `MessageElement` to generate number for a new object, as shown in the following syntax:

```
SaveAsObjectRequestType saveAsObjectRequestType = new SaveAsObjectRequestType();
AgileSaveAsObjectRequestType agileSaveAsObjectRequestType[] = new
AgileSaveAsObjectRequestType[1];
agileSaveAsObjectRequestType[0].setParentClassIdentifier("objectName");
agileSaveAsObjectRequestType[0].setParentObjectNumber("objectNumber");
agileSaveAsObjectRequestType[0].setNewClassIdentifier("newObjectName");
    AgileRowType row = new AgileRowType();
    MessageElement messages[] = new MessageElement[1];
    String namespaceUri = null;
    messages[0] = new MessageElement(namespaceUri, "TagName");
    messages[0].addAttribute(namespaceUri, "attributeId", "Attribute ID");
    messages[0].addTextNode(newObjectNumber);
    row.set_any(messages);
    agileSaveAsObjectRequestType[0].setData(row);
```

You can also use `autoNumberSource` to generate number for a new object, using the following syntax:

```
SaveAsObjectRequestType saveAsObjectRequestType = new SaveAsObjectRequestType();
AgileSaveAsObjectRequestType agileSaveAsObjectRequestType[] = new
AgileSaveAsObjectRequestType[1];
agileSaveAsObjectRequestType[0].setParentClassIdentifier("objectName");
agileSaveAsObjectRequestType[0].setParentObjectNumber("objectNumber");
agileSaveAsObjectRequestType[0].setNewClassIdentifier("newObjectName");
agileSaveAsObjectRequestType[0].setAutoNumberSource("autoNumberSource");
```

---

**Note** See [Getting AutoNumbers](#) on page 29 for more details.

---

### Example: Saving a Part As a New Part

```
SaveAsObjectRequestType saveAsObjectRequestType = new SaveAsObjectRequestType();
AgileSaveAsObjectRequestType agileSaveAsObjectRequestType[] = new
AgileSaveAsObjectRequestType[1];
agileSaveAsObjectRequestType[0].setParentClassIdentifier("Part");
agileSaveAsObjectRequestType[0].setParentObjectNumber(partNumber1);
agileSaveAsObjectRequestType[0].setNewClassIdentifier("Part");
```

```
AgileRowType row = new AgileRowType();
MessageElement messages[] = new MessageElement[1];
String namespaceUri = null;
messages[0] = new MessageElement(namespaceUri, "Message_Num");
messages[0].addTextNode(newPartNumber);
row.set_any(messages);
agileSaveAsObjectRequestType[0].setData(row);
```

---

**Note** saveAs reference of the query cannot be in the same folder as the actual query. Multiple saveAs references cannot be created for the same query object in the same folder. While the first operation will work fine, the concurrent operations will fail.

---

## Special Handling in the saveAsObject Operation

In case of saving an object as a **Program**, you need to specify the **TemplateType** - *Active*, *Template* or *Proposed*. Optionally, you can also pass additional attributes, such as Scheduled Start Data, Tables to be copied, Apply to children, etc.

---

**Note** The default **TemplateType** of a Program is *Active*.

---

### Example: Saving an Object as a Program of type Template

```
SaveAsObjectRequestType saveAsObjectRequestType = new SaveAsObjectRequestType();
AgileSaveAsObjectRequestType agileSaveAsObjectRequestType[] = new
AgileSaveAsObjectRequestType[1];
agileSaveAsObjectRequestType[0] = new AgileSaveAsObjectRequestType();
agileSaveAsObjectRequestType[0].setParentClassIdentifier("Program");
agileSaveAsObjectRequestType[0].setParentObjectNumber( parentProgramNumber );
agileSaveAsObjectRequestType[0].setNewClassIdentifier("Program");
AgileRowType row = new AgileRowType();
MessageElement messages[] = new MessageElement[1];
String namespaceUri = null;
messages[0] = new MessageElement(namespaceUri, "name");
messages[0].addTextNode(newProgramNumber);
row.set_any(messages);
agileSaveAsObjectRequestType[0].setData(row);
PropertyType properties[] = new PropertyType[1];
properties[0] = new PropertyType();
properties[0].setPropertyName(
SchemaConstants.program_template.getValue() );
properties[0].setPropertyValue("Template");
agileSaveAsObjectRequestType[0].setOptions(properties);
String tables[] = {"PageTwo", "Team"};
agileSaveAsObjectRequestType[0].setTablesToCopy(tables);
agileSaveAsObjectRequestType[0].setApplyToChildren(true);
```

When you create a program, you can specify that it is a template by setting the value of the Template attribute to "Template". You can do this only when you create a program or when you save it as a new program. Existing programs cannot be changed from the "Active" or "Proposed" state to "Template".



## Deleting and Undeleting an Object

The deletion of an object in Agile is of two types - soft delete and hard delete.

With soft delete, which is carried out using the operation `deleteObject`, the object is marked as 'Deleted'. It is however not removed from the database, so that it can be restored using the operation `undeleteObject`. A soft-deleted object does not appear in search results, however with the operation `isDeletedObject` you can find these deleted objects.

With hard delete, the object is removed from the database permanently. These objects do not appear in search queries or pre-defined query results.

---

**Note** To delete and undelete an object, you must have Delete and Undelete privileges, respectively, for the particular object type. However, soft-deleted changes that have items on the Affected Items tab cannot be restored, regardless of the user's privileges.

---

Not all Agile PLM objects can be deleted. If you attempt to delete these objects, the `deleteObject` operation throws an exception. Also, if you try to delete an Item that is used on the BOM tab of another item, the Agile PLM server throws an exception.

Some of the objects that cannot be deleted are:

- An item with a pending change
- An item with a revision history
- An item with a canceled change
- A released change
- A manufacturer with one or more manufacturer parts
- A manufacturer part currently used on the Manufacturers tab of another object

### Example: Deleting a Part

```
DeleteObjectRequestType deleteObjectRequestType = new DeleteObjectRequestType();
AgileDeleteObjectRequest agileDeleteObjectRequest[] = new AgileDeleteObjectRequest[1];
agileDeleteObjectRequest[0] = new AgileDeleteObjectRequest();
agileDeleteObjectRequest[0].setClassIdentifier("Part");
agileDeleteObjectRequest[0].setObjectNumber(partNumber);
```

### Example: Undeleting a Part

```
UndeleteObjectRequestType undeleteObjectRequestType = new UndeleteObjectRequestType();
AgileUndeleteObjectRequest agileUndeleteObjectRequest[] = new
AgileUndeleteObjectRequest[1];
agileUndeleteObjectRequest[0] = new AgileUndeleteObjectRequest();
agileUndeleteObjectRequest[0].setClassIdentifier("Part");
agileUndeleteObjectRequest[0].setObjectNumber(partNumber);
```

## Checking the Delete Status

You can verify whether an Object has been deleted or not by using the `isDeletedObject` operation.

### Example: Checking if a Part is deleted

```
IsDeletedObjectRequestType isDeletedObjectRequestType = new IsDeletedObjectRequestType();
AgileIsDeletedObjectRequest agileIsDeletedObjectRequest[] = new
AgileIsDeletedObjectRequest[1];
agileIsDeletedObjectRequest[0] = new AgileIsDeletedObjectRequest();
agileIsDeletedObjectRequest[0].setClassIdentifier("Part");
agileIsDeletedObjectRequest[0].setObjectNumber(partNumber);
```

## Updating an Object

You can update any object with the operation `updateObject` by setting the values of the desired attributes.

### Example: Updating a Part

```
UpdateObjectRequestType updateObjectRequestType = new UpdateObjectRequestType();
AgileUpdateObjectRequest agileUpdateObjectRequest[] = new AgileUpdateObjectRequest[1];
agileUpdateObjectRequest[0] = new AgileUpdateObjectRequest();
agileUpdateObjectRequest[0].setClassIdentifier("Part");
agileUpdateObjectRequest[0].setObjectNumber(partNumber);
    AgileRowType rows = new AgileRowType();
    MessageElement messages[] = new MessageElement[1];
    String namespaceUri = null;
    messages[0] = new MessageElement(namespaceUri, "Message_Desc");
    messages[0].addTextNode("Updated value of Doc Description");
    rows.set_any(messages);
    agileUpdateObjectRequest[0].setData(rows);
```

## Getting the Status of an Object

In a workflow or a lifecycle, a routable object passes through various states. Subsequent action on this object requires ascertaining its current state, the states it has already crossed and the states it must go through. This information is obtained using the operation `getStatus`.

The response object will consist of `AgileStatusType` objects for `nextDefaultStatus`, `nextValidStatuses`, `currentStatus`. For complete details, refer Schema Documentation at Oracle eDelivery Site.

### Example: Getting the status of an ECO

```
GetStatusRequestType getStatusRequestType = new GetStatusRequestType();
AgileGetStatusRequestType agileGetStatusRequestType[] = new AgileGetStatusRequestType[1];
agileGetStatusRequestType[0] = new AgileGetStatusRequestType();
agileGetStatusRequestType[0].setClassIdentifier("ECO");
agileGetStatusRequestType[0].setObjectNumber( changeNumber );
```

## Getting the AutoNumbers

An AutoNumber source is a predefined sequence of numbers that automatically assign a number to an object. An Agile PLM class can have one or more AutoNumber sources. These are defined in the Admin node of the Agile Java Client.

To get a 'next in sequence' AutoNumber, specify the `autoNumberSource` and `objectType` attributes in the operation `getAutoNumbers`.

---

**Note** The Manufacturers and Manufacturer Parts classes, and their user-defined subclasses, do not support automatic numbering.

---

### Example: Getting Autonumbers for Part Class

```
GetAutoNumbersRequestType getAutoNumbersRequestType = new GetAutoNumbersRequestType();
AgileGetAutoNumbersRequestType agileGetAutoNumbersRequestType[] = new
AgileGetAutoNumbersRequestType[2];
agileGetAutoNumbersRequestType[0].setClassIdentifier("Part");
agileGetAutoNumbersRequestType[0].setAutoNumberIdentifier( new
String[]{"PartNumber"} );
agileGetAutoNumbersRequestType[0].setSize(3);
    agileGetAutoNumbersRequestType[1].setClassIdentifier("ECO");
    agileGetAutoNumbersRequestType[1].setIncludeAllAutoNumberSource(true);
    agileGetAutoNumbersRequestType[1].setSize(2);
getAutoNumbersRequestType.setRequests(agileGetAutoNumbersRequestType);
```

## Getting all the Classes

You can retrieve the classes for each object type with the operation `getAllClasses`. Your program can then provide a method to pick the desired class from the list.

You can specify the `ClassFilterType` as follows:

- |                                 |   |
|---------------------------------|---|
| <b>ClassFilterType.ALL</b>      | - To retrieve all the classes and their subclasses. |
| <b>ClassFilterType.TOP</b>      | - To retrieve all the classes only                  |
| <b>ClassFilterType.CONCRETE</b> | - To retrieve all the subclasses only               |

The syntax for this operation is:

```
GetAllClassesRequestType getAllClassesRequestType = new GetAllClassesRequestType();
getAllClassesRequestType.setLevel(ClassFilterType.ALL);
```

## Getting the Subclasses of a Class

Although you can retrieve all Subclasses by using the operation `getAllClasses` with `ClassFilterType` filter, you may require Subclasses of a particular Class. This can be achieved by using the operation `getSubClasses`, in which, you can specify the Agile API name of the desired

Class.

ClassType objects are obtained from the response.

**Example: Getting all Subclasses of the Class 'Changes'**

```
GetSubClassesRequestType getSubClassesRequestType = new GetSubClassesRequestType();
AgileGetSubClassesRequestType agileGetSubClassesRequestType[] = new
AgileGetSubClassesRequestType[1];
agileGetSubClassesRequestType[0].setClassIdentifier("Changes");
```

## Getting Agile Classes

```
GetAgileClassRequestType getAgileClassRequestType =
    new GetAgileClassRequestType();
AgileGetClassRequestType[] agileGetClassRequestTypeArray =
    new AgileGetClassRequestType[1];
agileGetClassRequestTypeArray[0] = new AgileGetClassRequestType();
agileGetClassRequestTypeArray[0].setClassIdentifier("Part"); //API name of the class
getAgileClassRequestType.setRequests(agileGetClassRequestTypeArray);
```

# Subscribing to Agile PLM Objects

**This chapter includes the following:**

---

▪ Subscribe Privilege .....	31
▪ Subscription Notifications .....	31
▪ Getting Subscriptions for an Object .....	32
▪ Modifying the Subscriptions for an Object .....	32

When you load an Agile PLM business object, such as an item or change, you can then subscribe to that object. Once you subscribe to the object, you will receive a Notification whenever a triggering event occurs for that object. You can specify which events trigger a Notification. Subscription events can be a lifecycle change, a change to attachment files, or a change to the value of any cell that is made available for subscription.

You can subscribe to both routable and non-routable objects. The Agile web services provide an interface which enables retrieving and modifying all subscriptions for an object. All objects that a user has subscribed to are listed on the user's Subscription table.

## Subscribe Privilege

To subscribe to an object, you must have the Subscribe privilege for that class. Many predefined Agile PLM roles, such as Creator, already have the Subscribe privilege for several object classes. To change your roles and privileges, see the administrator of your Agile PLM system.

## Subscription Notifications

Subscription events trigger two types of Agile PLM Notifications:

- **Email** – Email Notifications are sent only if the user's Receive Email Notification preference is set to **Yes**. For information on user and system preferences, refer to *Agile PLM Administrator Guide*.
- **Inbox** – Inbox Notifications occur automatically regardless of user preferences

A user with Administrator privileges can create and configure these Notifications in Java Client which provides two very similar dialogs for this purpose. The reason for the two dialogs is due to the fact that there are two sets of Email and Inbox Notifications:

- Those that the "To" field is grayed out
- Those that the administrator user can select recipients who are notified when the subscription event is triggered

## Getting Subscriptions for an Object

To retrieve the current subscriptions for an object, use `getSubscriptions` method, which returns an array of all subscription objects, both enabled and disabled.

The following example shows how to get subscriptions for an object.

**Example: Getting subscriptions for an object**

## Modifying the Subscriptions for an Object

You can use the Agile web services to modify subscriptions for the current user only. If you change your subscriptions for a particular business object, other users' subscriptions for that object remain unaffected.

The list of subscription events for any object is set at the server and cannot be modified by the Agile web service. However, you can select the fields (attributes) you want subscribed. If you have Administrator privileges, you can also modify classes to define which fields are available for subscription.

To work with a subscription, use the following `modifySubscriptions` operation.

## Working with Tables

This chapter includes the following:

- Operations Supported on Tables..... 33
- Loading a Table ..... 35
- Working with the Readonly Tables ..... 37
- Retrieving the Metadata of a Table..... 37
- Adding Rows to a Table..... 37
- Updating Rows in a Table..... 43
- Removing Rows from a Table ..... 44
- Clearing a Table ..... 44
- Copying Tables..... 45
- Redlining a Table..... 45

This chapter describes how to work with the Agile PLM Table and provides sample code snippets.

### About Tables

Agile data is contained in tables. In Agile Web Client, these tables are equivalent to the separate tabs in a window, such as the Manufacturers and BOM tabs.

The Agile Web Services do not support random access of rows to a table. This means that you cannot retrieve a specific row by index number and then update it.

### Operations Supported on Tables

Web Services supports table operations for Agile PLM's PC and PQM solutions.

#### PC

Table Name	Objects	Web Services API
Item Changes Pending Changes	Item	loadTable, isReadOnlyTable
Item Changes Change History	Item	loadTable, isReadOnlyTable
Item BOM	Item	loadTable, copyTable, clearTable, isReadOnlyTable, addRows, removeRows, updateRows
Item Manufacturers	Item	loadTable, copyTable, clearTable, isReadOnlyTable, addRows, removeRows, updateRows

Table Name	Objects	Web Services API
Item Sites	Item	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Item Prices	Item	loadTable, isReadonlyTable
Item Quality	Item	loadTable, isReadonlyTable
Item Compliance Compositions	Item	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Item Compliance Substances	Item	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Item Compliance Specifications	Item	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Item Relationships	Item	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Item Where Used	Item	loadTable, isReadonlyTable
Changes Affected Items Table	Changes	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Changes AI Redline Title Block	Changes	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Changes AI Redline BOM	Changes	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Changes AI Redline Manufacturers	Changes	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Changes AI Redline Attachments	Changes	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Changes Relationships	Changes	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Mfrs Relationships	Mfrs	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Mfrs Where Used	Mfrs	loadTable, isReadonlyTable
Mfr Parts Prices	Mfrs	loadTable, isReadonlyTable
Mfr Parts Compliance Compositions	Mfr Parts	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Mfr Parts Compliance Substances	Mfr Parts	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Mfr Parts Compliance Specifications	Mfr Parts	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Mfr Parts Suppliers	Mfr Parts	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows



Table Name	Objects	Web Services API
Mfr Parts Relationships	Mfr Parts	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Mfr Parts Where Used	Mfr Parts	loadTable, isReadonlyTable
Sites Relationships	Sites	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
Sites History	Sites	loadTable, isReadonlyTable

**PQM**

Table Name	Objects	Web Services API
PSR Affected Items	PSR	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
PSR Related PSR	PSR	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
PSR Relationships	PSR	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
QCR Affected Items	QCR	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows
QCR Relationships	QCR	loadTable, copyTable, clearTable, isReadonlyTable, addRows, removeRows, updateRows

## Loading a Table

You can use the operation `loadTable` to load a table from Agile PLM system. This operation takes `tablesIdentifier` parameter along with `classIdentifier` and `objectIdentifier`.

Tables vary for each Agile PLM dataobject. Tables for change objects are different from tables for items. Each table for a particular dataobject is identified by a constant in the constants class or by the API name for that dataobject. Item constants are contained in the `ItemConstants` class, change constants are contained in the `ChangeConstants` class, and so on.

**Example: Loading the Table of a Part**

```
RequestTableType table[] = new RequestTableType[1];
table[0] = new RequestTableType();
table[0].setClassIdentifier("Part");
table[0].setObjectNumber( partNumber );
table[0].setTableIdentifier("table01");
loadTableRequestType.setTableRequest(table);
```

## Special Handling in the loadTable Operation

### Example: Loading a Table for an Object Version

```
table[0] = new RequestTableType();
table[0].setClassIdentifier("FileFolder");
table[0].setObjectNumber( folderNumber );
table[0].setTableIdentifier("Files");
PropertyType properties[] = new PropertyType[1];
properties[0] = new PropertyType();
properties[0].setPropertyName( SchemaConstants.folderVersion.getValue() );
properties[0].setPropertyValue( folderVersion );
table[0].setOptions(properties);
loadTableRequestType.setTableRequest(table);
```

### Example: Loading a Table for an Object Revision

```
table[0] = new RequestTableType();
table[0].setClassIdentifier( "Part" );
table[0].setObjectNumber( partNumber );
table[0].setTableIdentifier("TitleBlock");
PropertyType properties[] = new PropertyType[1];
properties[0] = new PropertyType();
properties[0].setPropertyName( SchemaConstants.revision.getValue() );
properties[0].setPropertyValue( partVersion );
table[0].setOptions(properties);
loadTableRequestType.setTableRequest(table);
```

### Example: Loading a Table for a Redline Change

```
table[0] = new RequestTableType();
table[0].setClassIdentifier( "Part" );
table[0].setObjectNumber( partNumber );
table[0].setTableIdentifier("TitleBlock");
PropertyType properties[] = new PropertyType[1];
properties[0] = new PropertyType();
properties[0].setPropertyName(SchemaConstants.redline_change.getValue() );
properties[0].setPropertyValue( changeNumber );
table[0].setOptions(properties);
loadTableRequestType.setTableRequest(table);
```

### Example: Loading a Table for a Site Object

```
table[0] = new RequestTableType();
table[0].setClassIdentifier( "Part" );
table[0].setObjectNumber( parentPartNumber );
table[0].setTableIdentifier("BOM" );
PropertyType properties[] = new PropertyType[1];
properties[0] = new PropertyType();
properties[0].setPropertyName( SchemaConstants.site.getValue() );
properties[0].setPropertyValue( sitel );
```

```
table[0].setOptions(properties);
loadTableRequestType.setTableRequest(table);
```

## Working with the Readonly Tables

Several Agile PLM tables store history information or data about related objects. These tables are read-only and as such, you cannot modify these tables. When you write code to access a table, use the operation `isReadOnlyTable` to check if the table is read-only.

## Retrieving the Metadata of a Table

You may require the metadata information of a table, which is the underlying data that describes a table's properties. This is useful when you need to identify the attributes of a particular table, its ID, or its table name without having to load a dataobject. The metadata is obtained in the form of `AttributeType` objects from the response.

For this, use the operation `getTableMetadata` specifying the `tableIdentifier` and `classIdentifier`.

### Example: Retrieving Metadata of a Table

```
agileGetTableMetadataRequestType[0].setClassIdentifier("Part");
agileGetTableMetadataRequestType[0].setTableIdentifier("table01");
getTableMetadataRequestType.setRequests(agileGetTableMetadataRequestType);
```

## Adding Rows to a Table

To create a table row, use the operation `addRows`, which creates a new row and initializes it with the data specified in the `rows` parameter. The `rows` parameter of `addRows` is available to pass the following data:

- a set of attributes and values for the row's cells
- an Agile PLM object (such as an Item) to add to the table

When you add a row to a table, it is not necessarily added at the end of the table.

---

**Note** You cannot add an empty row to a table.

---

### Example: Adding Rows in a BOM Table

With the `addRows` operation, you can add a child element to a Part by adding rows to the BOM table of the parent object.

```
RequestTableType table = new RequestTableType();
table.setClassIdentifier("Part");
table.setObjectNumber(parentPartNumber);
table.setTableIdentifier("BOMtable_API_Name");
AgileRowType[] rows = new AgileRowType[1];
```

```
rows[0] = new AgileRowType();
String namespaceUri = null;
MessageElement messages[] = new MessageElement[1];
rows[0].set_any(messages);
messages[0] = new MessageElement(namespaceUri, "BOM_Child_Number");
messages[0].addAttribute(namespaceUri, "itemNumber");
messages[0].addTextNode( BOMchildPartNumber );
agileAddRowsRequest[0].setRow(rows);
agileAddRowsRequest[0].setObjectInfo(table);
addRowsRequestType.setData(agileAddRowsRequest);
```

## Special Handling in the addRows Operation

---

**Note** All additional attributes like revision, site etc should be passed as options. Site and Revision should be passed along the individual row.

---

### Adding a Site to the Sites Tab of an Item

```
final String COMMONNAMESPACEURI = "http://xmlns.oracle.com/AgileObjects/Core/Common/V1";
final String XSIPREFIX = org.apache.axis.Constants.NS_PREFIX_SCHEMA_XSI;
final String TYPE = SchemaConstants.type.getValue();
AgileListEntryType lst03 = new AgileListEntryType();
SelectionType[] multiSelect = new SelectionType[1];
multiSelect[0] = new SelectionType();
multiSelect[0].setValue("Bangalore");
lst03.setSelection(multiSelect);
MessageElement dataCell = new MessageElement(namespaceUri, "siteName");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, TYPE, "AgileListEntryType");
dataCell.setObjectValue(lst03);
message[0] = dataCell;
row[0].set_any(message);
```

### Adding Suppliers to the Suppliers Tab of an Item

You can add suppliers to the suppliers tab of an item using the following two methods:

#### Method 1

```
final String COMMONNAMESPACEURI = "http://xmlns.oracle.com/AgileObjects/Core/Common/V1";
final String XSIPREFIX = org.apache.axis.Constants.NS_PREFIX_SCHEMA_XSI;
final String TYPE = SchemaConstants.type.getValue();
ObjectReferentIdType multiSelect = new ObjectReferentIdType();
multiSelect.setClassIdentifier("Broker");
multiSelect.setObjectIdentifier("SAP0265");
MessageElement dataCell = new MessageElement(namespaceUri, "supplier01");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, TYPE,
"ObjectReferentIdType");
```

```

dataCell.setObjectValue(multiSelect);
message[0] = dataCell;
row[0].set_any(message);

```

## Method 2

```

final String COMMONNAMESPACEURI = "http://xmlns.oracle.com/AgileObjects/Core/Common/V1";
final String XSIPREFIX = org.apache.axis.Constants.NS_PREFIX_SCHEMA_XSI;
final String TYPE = SchemaConstants.type.getValue();
    AgileObjectListEntryType multilist01 = new AgileObjectListEntryType();
    ObjectReferentIdType[] obj = new ObjectReferentIdType[1];
    obj[0] = new ObjectReferentIdType();
    obj[0].setClassIdentifier("Broker");
    obj[0].setObjectIdentifier("SAP0265");
    multilist01.setSelection(obj);
    MessageElement dataCell = new MessageElement(namespaceUri, "supplier");
    dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, TYPE,
        "AgileObjectListEntryType");
    dataCell.setObjectValue(multilist01);
    message[0] = dataCell;
    row[0].set_any(message);

```

## Adding Suppliers to a Manufacturer Part

```

final String COMMONNAMESPACEURI = "http://xmlns.oracle.com/AgileObjects/Core/Common/V1";
final String XSIPREFIX = org.apache.axis.Constants.NS_PREFIX_SCHEMA_XSI;
final String TYPE = SchemaConstants.type.getValue();
    RequestTableType objectInfo = new RequestTableType();
    objectInfo.setClassIdentifier(subclassId);
    objectInfo.setObjectNumber(objectNumber);
    objectInfo.setTableIdentifier("tableId");
    agileAddRowsRequests[0].setObjectInfo(objectInfo);
    PropertyType[] options = new PropertyType[1];
    options[0] = new PropertyType();
    options[0].setPropertyName(SchemaConstants.manufacturer_name.getValue());
    options[0].setPropertyValue("Cisco");
        agileAddRowsRequests[0].setOptions(options);
        AgileAddRowsRequest[] agileAddRowsRequests = new AgileAddRowsRequest[1];
        agileAddRowsRequests[0] = new AgileAddRowsRequest();
        AgileRowType[] row = new AgileRowType[1];
        row[0] = new AgileRowType();
        agileAddRowsRequest.setRow(row);
        String namespaceUri = null;
    MessageElement[] message = new MessageElement[1];
    AgileObjectListEntryType multilist01 = new AgileObjectListEntryType();
    ObjectReferentIdType[] obj = new ObjectReferentIdType[1];
    obj[0] = new ObjectReferentIdType();
    obj[0].setClassIdentifier("Broker");
    obj[0].setObjectIdentifier("SAP0265");
    multilist01.setSelection(obj);
    MessageElement dataCell = new MessageElement(namespaceUri, "supplier");

```

```
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, TYPE,
"AgileObjectListEntryType");
dataCell.setObjectValue(multilist01);
message[0] = dataCell;
row[0].set_any(message);
```

## Adding Manufacturer Part to AML of an Item

You can add a Manufacturer Part to the AML of an Item using the following two methods:

### Method 1

```
MessageElement[] message = new MessageElement[2];
MessageElement dataCell = new MessageElement(namespaceUri, "mfrPartNumber");
dataCell.setObjectValue("bosco");
message[0] = dataCell;
MessageElement dataCell = new MessageElement(namespaceUri, "mfrName");
dataCell.setObjectValue("cisco");
message[1] = dataCell;
row[0].set_any(message);
```

### Method 2

```
MessageElement[] message = new MessageElement[1];
ObjectReferentIdType obj = new ObjectReferentIdType();
obj.setClassIdentifier("ManufacturerPart");
obj.setObjectIdentifier("MfrP_01");
PropertyType[] options = new PropertyType[1];
options[0] = new PropertyType();
options[0].setPropertyName(SchemaConstants.manufacturer_name.getValue());
options[0].setPropertyValue("Manu_4570");
obj.setOptions(options);
MessageElement dataCell = new MessageElement(namespaceUri, "mfrPartNumber");
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, TYPE, "ObjectReferentIdType");
dataCell.setObjectValue(obj);
message[0] = dataCell;
row[0].set_any(message);
```

## Adding Manufacturer Part to the Relationships Tab

```
ObjectReferentIdType obj = new ObjectReferentIdType();
obj.setClassIdentifier("ManufacturerPart");
obj.setObjectIdentifier("m12444");
PropertyType[] options = new PropertyType[1];
options[0] = new PropertyType();
options[0].setPropertyName(SchemaConstants.manufacturer_name.getValue());
options[0].setPropertyValue("Cisco");
obj.setOptions(options);
MessageElement dataCell = new MessageElement(namespaceUri, "name");
```

```

dataCell1.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, TYPE,
"ObjectReferentIdType");
dataCell1.setObjectValue(obj);
message[0] = dataCell1;
row[0].set_any(message);

```

## Adding Affected Item to a Change

```

MessageElement dataCell11 = new MessageElement(namespaceUri, "itemNumber");
dataCell11.setObjectValue("P00400");
message[0] = dataCell11;

MessageElement dataCell12 = new MessageElement(namespaceUri, "effectiveDate");
dataCell12.setObjectValue(new Date());
dataCell12.addAttribute(XSIPREFIX, Constants.URI_DEFAULT_SCHEMA_XSD, "type",
"dateTime");
message[1] = dataCell12;
MessageElement dataCell13 = new MessageElement(namespaceUri, "newRev");
dataCell13.setObjectValue("Item_01");
dataCell13.addAttribute(XSIPREFIX, Constants.URI_DEFAULT_SCHEMA_XSD, TYPE, "string");
message[2] = dataCell13;

row[0].set_any(message);

```

## Adding Site Specific Item to the BOM Tab

To add a child object to a specific site, we can utilize either the `setOptions` feature on the table object, or use the `setAdditionalRowInfo` method on the row object. Using `setOptions` on the table object will add all new rows to a particular site. On the other hand, `setAdditionalRowInfo` may be used to specify a site for each individual row, meaning that if several rows are to be added with a web service call, each row may be added to a different site.

### Example: Adding Site Specific Item to the BOM Tab

In this example, `setAdditionalRowInfo` is used to add a given row to a specific site of a Part using the operation `addRows`.

```

RequestTableType table = new RequestTableType();
table.setClassIdentifier("Part");
table.setObjectNumber( parentPartNumber );
table.setTableIdentifier( "BOM" );
AgileRowType[] rows = new AgileRowType[1];
rows[0] = new AgileRowType();
String namespaceUri = null;
MessageElement messages[] = new MessageElement[1];
    rows[0].set_any(messages);
    messages[0] = new MessageElement(namespaceUri, "itemNumber");
    messages[0].addTextNode( childPartNumber );
    AdditionalInfoType additionalInfoType = new AdditionalInfoType();
    additionalInfoType.setSite(site1);
    rows[0].setAdditionalRowInfo(additionalInfoType);
    agileAddRowsRequest[0].setRow(rows);

```

```
agileAddRowsRequest[0].setObjectInfo(table);
addRowsRequestType.setData(agileAddRowsRequest);
```

## Adding Site Specific AML to the Manufacturers Tab

For adding a Manufacturer to a Part at a specific site, use the `setOptions` feature by providing a name-value pair using which a particular site is identified. Subsequently, the web service adds the manufacturer to the site as specified in the options.

A manufacturer part is specified through a `MessageElement` in the operation `addRows`, the message element cannot be specified in the usual manner. In this case, the message element for the manufacturer part must be of type `ObjectReferentIdType`. Consequently, an object identifier type object is created and appropriate class and object identifier values are set, using the manufacturer part class and its number, respectively.

```
RequestTableType table = new RequestTableType();
table.setClassIdentifier("Part");
table.setObjectNumber( partNumber );
table.setTableIdentifier( "Manufacturers" );
PropertyType[] properties = new PropertyType[1];
properties[0] = new PropertyType();
properties[0].setPropertyName( SchemaConstants.site.getValue() );
properties[0].setPropertyValue( site1 );
table.setOptions(properties);
AgileRowType[] rows = new AgileRowType[1];
rows[0] = new AgileRowType();

String namespaceUri = null;
MessageElement messages[] = new MessageElement[1];
messages[0] = new MessageElement(namespaceUri, "mfrPartNumber");
ObjectReferentIdType objRefId = new ObjectReferentIdType();
objRefId.setClassIdentifier( "ManufacturerPart" );
objRefId.setObjectIdentifier( manufPartNumber );
messages[0].addAttribute(Constants.NS_PREFIX_SCHEMA_XSI, COMMONNAMESPACEURI, "type",
"ObjectReferentIdType");
PropertyType[] properties_manufName = new PropertyType[1];
properties_manufName[0] = new PropertyType();
properties_manufName[0].setPropertyName(
SchemaConstants.manufacturer_name.getValue() );
properties_manufName[0].setPropertyValue( manufName );
objRefId.setOptions(properties_manufName);
messages[0].setObjectValue(objRefId);

rows[0].set_any(messages);
agileAddRowsRequest[0].setRow(rows);
agileAddRowsRequest[0].setObjectInfo(table);
addRowsRequestType.setData(agileAddRowsRequest);
```



## Adding Material to Item table

You are required to add the *Substance* type (or Class) also in the **addRows** operation request. If you do not specify a type, the application assumes you are trying to add an 'object' of type *Substance*. Since the object being added is of the type 'Material', the server will reuse it as Substance. Hence, you are required to specify the type of the object you are adding. The valid values are *Substance* and *Substance Group*.

```
public static class AddRowsDataToIPC17521DeclarationItemComposition {
    public static String methodName = "runPositive" ;
    public static String [] paramsOrder = {"classIds", "objectNumbers", "tableIds",
"attrsSet"} ;

    public static Integer [] classIds = {2000005975};
    public static String [] objectNumbers = {"IPC17521DEC_ANY"} ;
    public static Integer [] tableIds = {2000002780} ;
    public static Object [][][] attrsSet = {
        {
            {
                {"substanceName", 2000002745, "MATERIAL_ANY", "STANDARD"},
                {"itemNumber", 2000002974, "TEST_PART_ANY", "STANDARD"},
            }
        }
    };
}
```

## Updating Rows in a Table

Rows in a table are updated using the operation **updateRows**.

In the following example, the **rowID** is set after performing the operation **loadTable** and getting the **rowID** from the response.

```
UpdateRowsRequestType updateRowsRequestType = new UpdateRowsRequestType();
AgileUpdateRowsRequest agileUpdateRowsRequest[] = new AgileUpdateRowsRequest[1];
agileUpdateRowsRequest[0] = new AgileUpdateRowsRequest();
RequestTableType table = new RequestTableType();
table.setClassIdentifier( "ECO" );
table.setObjectNumber( changeNumber );
table.setTableIdentifier("AffectedItems" );
AgileUpdateRow updateRow[] = new AgileUpdateRow[1];
updateRow[0] = new AgileUpdateRow();
updateRow[0].setRowId(getRowID("ECO", changeNumber, "AffectedItems", partNumber )
);
AgileRowType row = new AgileRowType();
String namespaceUri = null;
MessageElement messages[] = new MessageElement[1];
Date date = new Date();
date.setTime( date.getTime() );
messages[0] = new MessageElement(namespaceUri, "effectiveDate");
messages[0].addAttribute("date_px", Constants.URI_DEFAULT_SCHEMA_XSD, "type",
"dateTime");
messages[0].setObjectValue(date);
```

```
row.set_any(messages);
updateRow[0].setRow(row);
agileUpdateRowsRequest[0].setRow(updateRow);
agileUpdateRowsRequest[0].setObjectInfo(table);
updateRowsRequestType.setData(agileUpdateRowsRequest);
```

---

**Note** See [Appendix](#) on page 281 for [getRowId](#) on page 285 helper method.

---

## Removing Rows from a Table

To remove a row from a table, use the operation `removeRows` operation, which requires `tableIdentifier`, `rowID`, besides `objectIdentifier`, `objectNumber`, and `objectInfo`.

If a table is read-only, you can't remove rows from it. To check the read/write status of a tables, see [Working with ReadOnly Tables](#) on page 37.

If you are working with a released revision of an item, you cannot remove a row from the item's tables until you create a change order for a new revision.

### Example: Removing a Table Row

```
RemoveRowsRequestType removeRowsRequestType = new RemoveRowsRequestType();
AgileRemoveRowsRequest agileRemoveRowsRequest[] = new AgileRemoveRowsRequest[1];
agileRemoveRowsRequest[0] = new AgileRemoveRowsRequest();
RequestTableType table = new RequestTableType();
table.setClassIdentifier("Part");
table.setObjectNumber( parentPartNumber );
table.setTableIdentifier( ItemConstants.TABLE_BOM.toString() );
agileRemoveRowsRequest[0].setObjectInfo(table);
agileRemoveRowsRequest[0].setRowId( new Integer[] {getRowID("Part", parentPartNumber,
"BOM", childPartNumber)} );
removeRowsRequestType.setRows(agileRemoveRowsRequest);
```

---

**Note** See Appendix for the [getRowId helper method](#) on page 285

---

## Clearing a Table

You can clear the entire table by removing all the rows. This can be done by setting the `tableIdentifier` in the operation `clearTable`.

### Example: Clearing a Table

```
RequestTableType table1 = new RequestTableType();
table1.setClassIdentifier("Part");
table1.setObjectNumber( partNumber );
table1.setTableIdentifier( "tableAPIName" );
agileClearTableRequestType[0].setAgileTable(table1);
clearTableRequestType.setClearTable(agileClearTableRequestType);
```

## Copying Tables

You can copy all the rows of a table in an Agile object to another table by using the operation `copyTable`. This operation requires `classIdentifier`, `objectNumber` and `tableIdentifier`, and setting of the `SourceTable` and `TargetTable` values.

```
agileCopyTableRequestType[0] = new AgileCopyTableRequestType();
RequestTableType table1 = new RequestTableType();
RequestTableType table2 = new RequestTableType();
    table1.setClassIdentifier("Part");
    table1.setObjectNumber( partNumber1 );
    table1.setTableIdentifier("Compositions");
    table2.setClassIdentifier("Part");
    table2.setObjectNumber( partNumber2 );
    table2.setTableIdentifier( "Compositions" );
agileCopyTableRequestType[0].setSourceTable(table1);
agileCopyTableRequestType[0].setTargetTable(table2);
copyTableRequestType.setCopyTable(agileCopyTableRequestType);
```

## Redlining a Table

When you issue a change for a released item or a price agreement, the Agile Web Services lets you redline certain tables affected by the change. In the Agile PLM clients, redline tables visually identify values that have been modified from the previous revision. Red underlined text - thus the term “redline”, indicates values that have been added, and red strikethrough text indicates values that have been deleted. Those responsible for approving the change can review the redline data.

The Agile PLM system provides the following Redline tables:

- ▣ Redline BOM
- ▣ Redline Manufacturers (AML)
- ▣ Redline Price Lines
- ▣ Redline Title Block

### Example: Adding a Redlined BOM

```
RequestTableType table = new RequestTableType();
table.setClassIdentifier("Part");
table.setObjectNumber( parentPartNumber );
table.setTableIdentifier( "-803" );
ObjectReferentIdType multiSelect = new ObjectReferentIdType();
multiSelect.setClassIdentifier("Part");
multiSelect.setObjectIdentifier("P00407");
MessageElement dataCell = new MessageElement(namespaceUri, "itemNumber");
```

```
dataCell.addAttribute(XSIPREFIX, COMMONNAMESPACEURI, TYPE,  
"ObjectReferentIdType");  
dataCell.setObjectValue(multiSelect);  
message[0] = dataCell;  
row[0].set_any(message);  
PropertyType[] options = new PropertyType[1];  
options[0] = new PropertyType();  
options[0].setPropertyName(SchemaConstants.redline_change.getValue());  
options[0].setPropertyValue("C00644");  
row[0].setOptions(options);  
agileAddRowsRequest[0].setObjectInfo(table);
```

# Working with File Folders and Attachments

**This chapter includes the following:**

---

▪ Managing File Folders .....	47
▪ Getting a File from a File Folder .....	51
▪ Adding Files to a File Folder Object.....	53
▪ Managing Attachments .....	54

This chapter describes how to work with the Agile PLM File Folders and Attachments, and provides sample code snippets.

## Agile File Folders

A File Folder is a business object that specifies one or more files or URLs that are stored in the file PLM server vault. In addition, a file folder has its own set of tables. This means that you can create and load an independent file folder and add one or more files to its Files table. You can also search for a file folder, just as you would search for an Item or Change.

The File Folder Base Class has two Classes and each of these classes have their own respective Subclasses. This section describes File Folder features and components, and provides procedures to add, modify, or remove them.

## Managing File Folders

The Agile Web Services let you perform File Folder related tasks, such as:

- checking-in and checking-out files associated with the objects in the rows of an Attachments table
- adding files and URLs to an Attachments table
- deleting attachments

This section lists and describes these features, and provides necessary procedures to use the Agile Web Services to perform these tasks.

## Creating a File Folder

With the operation createObject, you can create a File Folder, as a file folder is an Agile Object.

### Example: Creating a File Folder

```
CreateObjectType createObjectType = new CreateObjectType();  
AgileCreateObjectRequest agileCreateObjectRequest[] = new AgileCreateObjectRequest[1];
```

```
agileCreateObjectRequest[0] = new AgileCreateObjectRequest();
agileCreateObjectRequest[0].setClassIdentifier("FileFolder");
AgileRowType row_1 = new AgileRowType();
MessageElement[] messages = new MessageElement[2];
String namespaceUri = null;
messages[0] = new MessageElement(namespaceUri, "number");
messages[0].addTextNode(folderNumber);
messages[1] = new MessageElement(namespaceUri, "description");
messages[1].addTextNode("File Folder Description");
row_1.set_any(messages);
agileCreateObjectRequest[0].setData(row_1);
createObjectRequestType.setRequests(agileCreateObjectRequest);
```

---

**Note** When you add a file or a URL to the row of the Attachments table of a business object, you will automatically create automatically a new file folder object that contains the associated file or URL.

---

#### Example: Creating a Design Object

```
CreateObjectRequestType createObjectRequestType = new CreateObjectRequestType();
AgileCreateObjectRequest agileCreateObjectRequest[] = new AgileCreateObjectRequest[1];
agileCreateObjectRequest[0] = new AgileCreateObjectRequest();
agileCreateObjectRequest[0].setClassIdentifier("Design");
AgileRowType row_1 = new AgileRowType();
MessageElement[] messages = new MessageElement[2];
String namespaceUri = null;
messages[0] = new MessageElement(namespaceUri, "number");
messages[0].addTextNode(designNumber);
messages[1] = new MessageElement(namespaceUri, "description");
messages[1].addTextNode("Design Desc");
row_1.set_any(messages);
agileCreateObjectRequest[0].setData(row_1);
createObjectRequestType.setRequests(agileCreateObjectRequest);
```

## Checking Out a File Folder

Before you can add, delete, or modify the files contained in a file folder, you must check out the file folder. With the appropriate privileges, you can check out a file folder as long as it is not already checked out by another user. Once a file folder is checked out, no one else can check it out or modify it.

The user who checked out a file folder, as well as other users who are change analysts or component engineers, can check it in. If the file folder was checked out to a location on the network, or to a shared drive or directory, anyone who has access to that network location or to that shared directory can check in the file folder.

Use the operation `checkOutFF` for checking out a file folder.

```
CheckOutFFRequestType checkOutFFRequestType = new CheckOutFFRequestType();
AgileCheckOutFFRequest agileCheckOutFFRequest[] = new AgileCheckOutFFRequest[1];
agileCheckOutFFRequest[0] = new AgileCheckOutFFRequest();
agileCheckOutFFRequest[0].setFolderNumber( folderNumber );
checkOutFFRequestType.setRequests(agileCheckOutFFRequest);
```

## Setting the Version of File Folder Files

A file folder can have several versions. When you add a file folder to the Attachments table of another business object, you can specify the file version to use. If you don't specify a file version, the Agile Web Services use the default or latest version. If you specify a file version, the row on the Attachments table is linked to that version.

If the parent object containing the Attachments table is an item, you can incorporate the item to lock the specified versions of its attachments.

### Example: Setting the version when adding a row to the Attachments table

This is carried out in two stages. First, add a row using the operation `addRows` to add a file folder and then update the table using the operation `updateRows`.

#### Stage 1 - Adding the file folder to the attachment table of a Part

```
AddRowsRequestType addRowsRequestType = new AddRowsRequestType();
AgileAddRowsRequest agileAddRowsRequest[] = new AgileAddRowsRequest[1];
agileAddRowsRequest[0] = new AgileAddRowsRequest();
RequestTableType table = new RequestTableType();
table.setClassIdentifier("Part");
table.setObjectNumber( parentPartNumber );
table.setTableIdentifier( "Attachments" );
folderNumber = "FOLDER00441";
    AgileRowType[] rows = new AgileRowType[1];
    rows[0] = new AgileRowType();
    String namespaceUri = null;
    String COMMONNAMESPACEURI = "http://xmlns.oracle.com/AgileObjects/Core/Common/V1";
    MessageElement messages[] = new MessageElement[1];
    messages[0] = new MessageElement(namespaceUri, "folderNumber");
    messages[0].addAttribute(Constants.NS_PREFIX_SCHEMA_XSI, COMMONNAMESPACEURI,
    "type", "ObjectReferentIdType");
    ObjectReferentIdType objRefId = new ObjectReferentIdType();
    objRefId.setClassIdentifier("FileFolder");
    objRefId.setObjectIdentifier( folderNumber );
    messages[0].setObjectValue(objRefId);
    rows[0].set_any(messages);
    agileAddRowsRequest[0].setRow(rows);
    agileAddRowsRequest[0].setObjectInfo(table);
addRowsRequestType.setData(agileAddRowsRequest);
```

#### Stage 2 - Updating the version of this newly created file folder on the attachment table

```
RequestTableType updateTable = new RequestTableType();
updateTable.setClassIdentifier( "Part" );
updateTable.setObjectNumber( parentPartNumber);
updateTable.setTableIdentifier( "Attachments" );
AgileUpdateRow updateRow[] = new AgileUpdateRow[1];
updateRow[0] = new AgileUpdateRow();
updateRow[0].setRowId(RowId);
AgileRowType row = new AgileRowType();
MessageElement messages_2[] = new MessageElement[1];
```

```
messages_2[0] = new MessageElement(namespaceUri, "folderVersion");
messages_2[0].addAttribute(Constants.NS_PREFIX_SCHEMA_XSI, COMMONNAMESPACEURI, "type",
"AgileListEntryType");
AgileListEntryType list = new AgileListEntryType();
    SelectionType[] selection = new SelectionType[1];
    selection[0] = new SelectionType();
    selection[0].setValue("2");
    list.setSelection(selection);
    messages_2[0].setObjectValue(list);
    row.set_any(messages_2);
    updateRow[0].setRow(row);

agileUpdateRowsRequest[0].setRow(updateRow);
agileUpdateRowsRequest[0].setObjectInfo(updateTable);
updateRowsRequestType.setData(agileUpdateRowsRequest);
```

## Cancelling a File Folder Checkout

If you check out a file folder and then decide not to modify it, or discard your changes and revert to the original file folder, you can cancel the checkout with the operation `cancelCheckoutFF`. When you cancel a checkout, the file folder is available for other users to check out.

---

**Note** Only the user who checked out a file folder can cancel the checkout.

---

### Example: Cancelling a file folder checkout

```
CancelCheckoutFFRequestType cancelCheckoutFFRequestType = new
CancelCheckoutFFRequestType();
AgileCancelCheckoutFFRequest agileCancelCheckoutFFRequest[] = new
AgileCancelCheckoutFFRequest[1];
agileCancelCheckoutFFRequest[0] = new AgileCancelCheckoutFFRequest();
agileCancelCheckoutFFRequest[0].setFolderNumber( folderNumber );
cancelCheckoutFFRequestType.setRequests( agileCancelCheckoutFFRequest );
```

## Checking In a File Folder

After you finish working on the file folder that you checked out, you can check it in using the operation `checkInFF`.

File folders can contain multiple files. When you check in a file folder, you automatically check in all the files that are contained in it. You do not need to specifically list the files contained in the file folder.

### Example: Checking In a File Folder

```
CheckInFFRequestType checkInFFRequestType = new CheckInFFRequestType();
AgileCheckInFFRequest agileCheckInFFRequest[] = new AgileCheckInFFRequest[1];
agileCheckInFFRequest[0] = new AgileCheckInFFRequest();
agileCheckInFFRequest[0].setFolderNumber( folderNumber );
checkInFFRequestType.setRequests( agileCheckInFFRequest );
```



## Deleting the File Folders

To delete a File Folder, use the operation `deleteObject`. You must have the Delete privilege for file folders to be able to delete them.

---

**Note** Deleting a file folder does not automatically remove its associated files from the file server. The Agile PLM administrator is responsible for purging deleted files.

---

To delete a row from the Attachments table of a business object, use the operation `removeRows`.

Removing a row from the Attachments table does not delete the associated file folder. You cannot delete a row from the Attachments table in the following situations:

- The parent object is an Item whose revision is incorporated.
- The selected attachment is currently checked out.

---

**Note** Only the user who checked out a file folder can cancel the checkout.

**Note** Deleting a file folder does not automatically remove its associated files from the file server. The Agile PLM administrator is responsible for purging the deleted files.

---

### Example: Deleting a File Folder

```
DeleteObjectRequestType deleteObjectRequestType = new DeleteObjectRequestType();
AgileDeleteObjectRequest agileDeleteObjectRequest[] = new AgileDeleteObjectRequest[1];
agileDeleteObjectRequest[0] = new AgileDeleteObjectRequest();
agileDeleteObjectRequest[0].setClassIdentifier("FileFolder");
agileDeleteObjectRequest[0].setObjectNumber(folderNumber);
deleteObjectRequestType.setRequests(agileDeleteObjectRequest);
```

## Getting a File from a File Folder

If a file folder is checked out by another user, you can still get a copy of the file folder file(s) and save it to your local machine. You can use the operation `getFileFF`, which returns the file stream associated with a row of the Attachments table. The file stream can be for one file or it can be a zipped file stream for multiple files, depending on how many files the associated file folder has.

If you call this operation from the file folder object, you return the zipped file stream for all files listed on the Files table. Whereas, if you call this operation from a row of the Files table of a file folder, you return a file stream for the specific file associated with that row.

```
GetFileFFRequestType getFileFFRequestType = new GetFileFFRequestType();
AgileGetFileFFRequest agileGetFileFFRequest[] = new AgileGetFileFFRequest[1];
agileGetFileFFRequest[0] = new AgileGetFileFFRequest();
agileGetFileFFRequest[0].setFolderNumber(folderNumber);
AgileFileAttachmentRequestType files[] = new AgileFileAttachmentRequestType[1];
files[0] = new AgileFileAttachmentRequestType();
files[0].setRowId(rowId);
agileGetFileFFRequest[0].setFiles(files);
getFileFFRequestType.setRequests(agileGetFileFFRequest);
```

**Example: Getting all the Files from a File Folder**

To get all the file of an Agile File Folder object, set `allFiles` variable to true.

```
GetFileFFRequestType getFileFFRequestType = new GetFileFFRequestType();
AgileGetFileFFRequest agileGetFileFFRequest[] = new AgileGetFileFFRequest[1];
agileGetFileFFRequest[0] = new AgileGetFileFFRequest();
agileGetFileFFRequest[0].setFolderNumber(folderNumber);
agileGetFileFFRequest[0].setAllFiles(true);
getFileFFRequestType.setRequests(agileGetFileFFRequest);
```

**Getting a File from a File Folder using a Download URL**

When you have to download extremely large files, you can use the operation `getFileFF` to generate a `downloadURL` for a file attachment present in an Agile File Folder. The request object contains a boolean to indicate that the download URL needs to be fetched and also the specifications that identify the attachment to be downloaded.

**Example: Getting a File using a URL**

```
GetFileFFRequestType getFileFFRequestType = new GetFileFFRequestType();
AgileGetFileFFRequest agileGetFileFFRequest[] = new AgileGetFileFFRequest[1];
agileGetFileFFRequest[0] = new AgileGetFileFFRequest();
agileGetFileFFRequest[0].setFolderNumber(folderNumber);
agileGetFileFFRequest[0].setDownloadUrl(true);
AgileFileAttachmentRequestType files[] = new AgileFileAttachmentRequestType[1];
files[0] = new AgileFileAttachmentRequestType();
files[0].setRowId(rowId);
agileGetFileFFRequest[0].setFiles(files);
getFileFFRequestType.setRequests(agileGetFileFFRequest);
```

**Getting a File from a particular Version of File Folder**

A file attachment can be downloaded from a particular version of an Agile File Folder Object. The request object contains the specifications that identify the attachment to be downloaded and the version of the folder, an array of bytes is obtained in the response object.

**Example: Getting a File from a Version of a File Folder**

```
GetFileFFRequestType getFileFFRequestType = new GetFileFFRequestType();
AgileGetFileFFRequest agileGetFileFFRequest[] = new AgileGetFileFFRequest[1];
agileGetFileFFRequest[0] = new AgileGetFileFFRequest();
agileGetFileFFRequest[0].setFolderNumber(folderNumber);
agileGetFileFFRequest[0].setFolderVersion(folderVersion);
AgileFileAttachmentRequestType files[] = new AgileFileAttachmentRequestType[1];
files[0] = new AgileFileAttachmentRequestType();
files[0].setRowId(rowId);
agileGetFileFFRequest[0].setFiles(files);
getFileFFRequestType.setRequests(agileGetFileFFRequest);
```

## Adding Files to a File Folder Object

With the operation `addFileFF`, you can add files to the 'Files' tab of a File Folder. Before you add the files, the File Folder must be checked out using the operation `checkOutFF`.

```
checkOutFolder(folderNumber);
agileAddFileFFRequestType[0].setFolderNumber(folderNumber);
AddFileFFType files[] = new AddFileFFType[1];
files[0] = new AddFileFFType();
files[0].setFileName("Filename.txt");
files[0].setDescription("Description for file");
files[0].setContent( "File Content...file".getBytes() );
agileAddFileFFRequestType[0].setFiles(files);
addFileFFRequestType.setRequest(agileAddFileFFRequestType);
```

### Example: Adding a File using its DFM Reference

For the files that were already added to DFM, you can add a file to the 'Files' tab of a file folder using a reference obtained from the DFM file server.

```
checkOutFolder(folderNumber);
agileAddFileFFRequestType[0].setFolderNumber(folderNumber);
AddFileReferenceFFType reference[] = new AddFileReferenceFFType[1];
reference[0] = new AddFileReferenceFFType();
reference[0].setFileId(fileId);
reference[0].setFileName("FileThrowReference.txt");
reference[0].setDescription("file added using a reference");
reference[0].setFileSize( new Long(1) );
agileAddFileFFRequestType[0].setFileRefs( reference );
addFileFFRequestType.setRequest(agileAddFileFFRequestType);
```

### Example: Adding URLs to a File Folder

```
agileAddFileFFRequestType[0].setFolderNumber(folderNumber);
AddUrlFFType urls[] = new AddUrlFFType[1];
urls[0] = new AddUrlFFType();
urls[0].setUrl("http://www.testurl_filefolder.com");
urls[0].setDescription("Test url description");
agileAddFileFFRequestType[0].setUrls(urls);
addFileFFRequestType.setRequest(agileAddFileFFRequestType);
```

## Adding Files in a File Folder

The Files table of a file folder lists the files and URLs associated with the object. To edit the table, you must first check out the file folder. You cannot add files or URLs to the Files table or delete them unless the file folder is checked out.

### Example: Adding a URL in a File Folder

Addition of URL attachments to the 'Files' tab of a file folder object can be carried out by specifying

the URL and folder number.

```
checkOutFolder(folderNumber);
agileAddFileFFRequestType[0].setFolderNumber(folderNumber);
AddUrlFFType urls[] = new AddUrlFFType[1];
urls[0] = new AddUrlFFType();
urls[0].setUrl("http://www.testurl_filefolder.com");
urls[0].setDescription("Test url description");
agileAddFileFFRequestType[0].setUrls(urls);
addFileFFRequestType.setRequest(agileAddFileFFRequestType);
```

## Managing Attachments

Attachments to objects contain information about the object or a manufacturing process. You can attach files and URLs by referencing them in a File Folder object. The File Folder object holds pertinent content, or Attachments. Most primary Agile API objects, such as Item, Change, Manufacturer, ManufacturerPart, Package, TransferOrder, User, and UserGroup, have an attachments table (or tab in the Java Client) that lists indirect references to the files or URLs that are in separate file folders. Each row in an Attachments table can refer to one file or to all files from a referenced file folder.

### Getting Attachments of an Object

A file attachment is retrieved using the operation `getFileAttachment`.

When the required file is present in a single, separate row, it is downloaded from the attachment tab using its `rowId`. In other cases, when there are several files in the same row and the desired file is one of them, the `fileId` must also be specified.

#### Example: Getting a single Attachment

```
agileGetFileAttachmentRequest[0].setClassIdentifier("Part");
agileGetFileAttachmentRequest[0].setObjectNumber(partNumber);
agileGetFileAttachmentRequest[0].setAllFiles(false);
AgileFileAttachmentRequestType attachments[] = new AgileFileAttachmentRequestType[1];
attachments[0] = new AgileFileAttachmentRequestType();
attachments[0].setRowId(rowId);
agileGetFileAttachmentRequest[0].setAttachments(attachments);
getFileAttachmentRequestType.setRequest(agileGetFileAttachmentRequest);
```

#### Example: Getting all the Attachments of an Object

To get all the attachments of an object, set `allFiles` to `True`.

```
agileGetFileAttachmentRequest[0].setClassIdentifier("Part");
agileGetFileAttachmentRequest[0].setObjectNumber(partNumber);
agileGetFileAttachmentRequest[0].setAllFiles(true);
AgileFileAttachmentRequestType attachments[] = new AgileFileAttachmentRequestType[1];
attachments[0] = new AgileFileAttachmentRequestType();
agileGetFileAttachmentRequest[0].setAttachments(attachments);
getFileAttachmentRequestType.setRequest(agileGetFileAttachmentRequest);
```

## Getting a Specific Attachment and a File Folder

When there are several files in the same row and the desired file is one of them, then a file can be downloaded from the attachment tab using its `fileId` along with the `rowId`. Using the `rowId` alone in this case is ineffective since all the files of that row will be obtained. To download only a specific file from such a set of files in a single row, the `fileId` is also needed.

For such cases, first set the `rowId` information, then obtain the `fileId` value and set the same into the 'files' element of the request object.

```
agileGetFileAttachmentRequest[0].setClassIdentifier("Part");
agileGetFileAttachmentRequest[0].setObjectNumber(partNumber);
agileGetFileAttachmentRequest[0].setAllFiles(false);
AgileFileAttachmentRequestType attachments[] = new AgileFileAttachmentRequestType[1];
attachments[0] = new AgileFileAttachmentRequestType();
attachments[0].setRowId(rowId);
int fileIds[] = new int[] { fileId1, fileId2 };
attachments[0].setFiles(fileIds);
agileGetFileAttachmentRequest[0].setAttachments(attachments);
GetFileAttachmentRequestType.setRequest(agileGetFileAttachmentRequest);
```

Instead of using the `fileId` for obtaining a file from a row with multiple files, you can also use the `rowId` of the files tab in the `fileFolder` object vis-a-vis the desired file.

To download only a specific file from a set of files in a single row in the Attachment tab, first set the `rowId` information, then obtain the `rowId` value of the corresponding `FileFolder` object of the file and set the same into the 'files' element of request object.

```
agileGetFileAttachmentRequest[0].setClassIdentifier("Part");
agileGetFileAttachmentRequest[0].setObjectNumber(partNumber);
agileGetFileAttachmentRequest[0].setAllFiles(false);
AgileFileAttachmentRequestType attachments[] = new AgileFileAttachmentRequestType[1];
attachments[0] = new AgileFileAttachmentRequestType();
attachments[0].setRowId(rowId);
int fileIds[] = new int[] { fileFolderRowId1, fileFolderRowId2 };
attachments[0].setFiles(fileIds);
agileGetFileAttachmentRequest[0].setAttachments(attachments);
GetFileAttachmentRequestType.setRequest(agileGetFileAttachmentRequest);
```

---

**Note** See [Appendix](#) on page 281 for sample helper methods.

---

## Getting a Specific Attachment using a URL

```
GetFileAttachmentRequestType GetFileAttachmentRequestType = new
GetFileAttachmentRequestType();
AgileGetFileAttachmentRequest agileGetFileAttachmentRequest[] = new
AgileGetFileAttachmentRequest[1];
agileGetFileAttachmentRequest[0] = new AgileGetFileAttachmentRequest();
agileGetFileAttachmentRequest[0].setClassIdentifier("Part");
```

```
agileGetFileAttachmentRequest[0].setObjectNumber(partNumber);
agileGetFileAttachmentRequest[0].setAllFiles(false);
agileGetFileAttachmentRequest[0].setDownloadUrl(true);
AgileFileAttachmentRequestType attachments[] = new AgileFileAttachmentRequestType[1];
attachments[0] = new AgileFileAttachmentRequestType();
attachments[0].setRowId(rowId);
agileGetFileAttachmentRequest[0].setAttachments(attachments);
GetFileAttachmentRequestType.setRequests(agileGetFileAttachmentRequest);
```

---

**Note** See [Appendix](#) on page 281 for sample helper methods.

---

## Adding Attachments to an Object

When you add a file or a URL to the Attachments table of a business object, the server automatically creates a new file folder containing the associated file or URL. The new row on the Attachments table references the new file folder. Use the operation `addFileAttachment` to add a File or a URL.

When you add a URL attachment, the server stores a reference to the Internet location but does not upload a file. Therefore, you cannot download a URL attachment. Agile Web Services validate URL strings that you attempt to check in as an attachment. If a URL is invalid, the Agile Web Services consider the string a filename instead of a URL.

You cannot add a file or URL to the Attachments table of an item if:

- The current revision has a pending or released MCO.
- The current revision is incorporated.

To add attachments, you have to specify the unique object to whose attachment tab the files will be added. Also, specify its class identifier and object number information.

The exact specification of the attachment to be added is defined as an object of type `AgileAddFileAttachmentRequestType`. This object includes information about the name of the file and its description and content.

### Example: Adding an Attachment to a Part

You can specify if multiple attachments should add into a single folder or multiple folders by supplying boolean value to `setSingleFolder`.

```
agileAddFileAttachmentRequest[0].setClassIdentifier("Part");
agileAddFileAttachmentRequest[0].setObjectNumber( partNumber );
AgileAddFileAttachmentRequestType attachments[] = new
AgileAddFileAttachmentRequestType[1];
attachments[0] = new AgileAddFileAttachmentRequestType();
attachments[0].setName("Filename.txt");
attachments[0].setDescription("Description for file ");
attachments[0].setContent( "File Content...file".getBytes() );
agileAddFileAttachmentRequest[0].setAttachments(attachments);
agileAddFileAttachmentRequest[0].setSingleFolder(false);
```

```
addFileAttachmentRequestType.setRequests(agileAddFileAttachmentRequest);
```

## Adding attachments by File Reference

You can add an attachment to an Agile object by using a file reference for a file that has already been added to DFM. This reference is obtained from the DFM file server.

### Example: Adding an Attachment to a Part using its File Reference

```
agileAddFileAttachmentRequest[0].setClassIdentifier("Part");
agileAddFileAttachmentRequest[0].setObjectNumber(partNumber);
AgileAddFileAttachmentReferenceRequestType reference[] = new
AgileAddFileAttachmentReferenceRequestType[1];
reference[0] = new AgileAddFileAttachmentReferenceRequestType();
reference[0].setFileId( DFMfileId );
reference[0].setFileName("FileThrowReference.txt");
reference[0].setDescription("File added using a reference");
reference[0].setFileSize( new Long(1) );
agileAddFileAttachmentRequest[0].setAttachmentRefs(reference);
addFileAttachmentRequestType.setRequests(agileAddFileAttachmentRequest);
```

## Adding Multiple Attachments into Single Folder

Addition of several file attachments to an Agile object requires explicit specification that all the files added to the Agile object must be added to a single folder. The element 'singleFolder' is used to specify that all the attachments added to the object must be added under a single folder.

### Example: Adding multiple Attachments into a single folder

```
AddFileAttachmentRequestType addFileAttachmentRequestType = new
AddFileAttachmentRequestType();
AgileAddFileAttachmentRequest agileAddFileAttachmentRequest[] = new
AgileAddFileAttachmentRequest[1];
agileAddFileAttachmentRequest[0] = new AgileAddFileAttachmentRequest();
agileAddFileAttachmentRequest[0].setClassIdentifier("Part");
agileAddFileAttachmentRequest[0].setObjectNumber(partNumber);
AgileAddFileAttachmentRequestType attachments[] = new
AgileAddFileAttachmentRequestType[2];
for(int i=0; i<attachments.length; i++){
    attachments[i] = new AgileAddFileAttachmentRequestType();
    attachments[i].setName("Filename" + (i+1) + ".txt");
    attachments[i].setDescription("Description for file " + (i+1) );
    attachments[i].setContent( ("File Content...file" + (i+1) ).getBytes() );
}
agileAddFileAttachmentRequest[0].setAttachments(attachments);
agileAddFileAttachmentRequest[0].setSingleFolder(true);
addFileAttachmentRequestType.setRequests(agileAddFileAttachmentRequest);
```

## Adding Files using SOAP Attachment

For adding very large attachments to an Object, use SOAP attachments.

While using SOAP attachments we create a datahandler to specify the file source and add the add the content as a soap attachment to the soap request as shown in the following example. Finally the `contentId` is set onto `AgileAddFileAttachmentRequestType`.

```
agileAddFileAttachmentRequest[0].setClassIdentifier("Part");
agileAddFileAttachmentRequest[0].setObjectNumber( partNumber );
AgileAddFileAttachmentRequestType attachments[] = new
AgileAddFileAttachmentRequestType[1];
attachments[0] = new AgileAddFileAttachmentRequestType();
attachments[0].setName("Filename.txt");
attachments[0].setDescription("Description for file ");

DataHandler dh = new DataHandler(new FileDataSource("c:\sample.txt" ));
AttachmentPart ap = new AttachmentPart(dh);
agileStub.addAttachment(ap);
attachments[0].setContentId(ap.getContentId());
agileAddFileAttachmentRequest[0].setAttachments(attachments);
agileAddFileAttachmentRequest[0].setSingleFolder(false);
addFileAttachmentRequestType.setRequests(agileAddFileAttachmentRequest);
```

---

**Note** This feature is supported by four operations - `addFileFF`, `addFileAttachment`, `checkInFF`, `checkInAttachment`.

**Note** See also - sample [AddFileSOAPAttachment](#) on page 283 method.

---

## Checking Out the Attachments

Checking out an Attachment process entails obtaining the file information and making changes to the same. The file information is also obtained in the response as an array of bytes. Checking out an attachment for any modifications is carried out with the `checkOutAttachment`.

### Example: Checking Out an Attachment of a Part

```
agileCheckOutAttachmentRequestType[0].setClassIdentifier("Part");
agileCheckOutAttachmentRequestType[0].setObjectNumber(partNumber);
CheckOutAttachmentType attachments[] = new CheckOutAttachmentType[1];
attachments[0] = new CheckOutAttachmentType();
attachments[0].setRowId(rowId);
agileCheckOutAttachmentRequestType[0].setAttachments(attachments);
checkOutAttachmentRequestType.setRequests(agileCheckOutAttachmentRequestType);
```



## Checking Out All the Attachments

You can check out all the attachments of an object by setting the boolean value of **allFiles** variable.

### Example: Checking Out all the Attachments of a Part

```
agileCheckOutAttachmentRequestType[0].setClassIdentifier("Part");
agileCheckOutAttachmentRequestType[0].setObjectNumber( partNumber );
agileCheckOutAttachmentRequestType[0].setAllFiles(true);
CheckOutAttachmentType attachments[] = new CheckOutAttachmentType[1];
attachments[0] = new CheckOutAttachmentType();
agileCheckOutAttachmentRequestType[0].setAttachments(attachments);
checkOutAttachmentRequestType.setRequests(agileCheckOutAttachmentRequestType);
```

## Checking Out Multiple Attachments from a Folder

When multiple files are associated with a single row, the attachment is identified by using its rowId in conjunction with the rowId of the attachment in the files tab of the file folder object. It requires usage of the rowId of the attachment along with its fileId to distinguish the attachment from all the other attachments in that row.

### Example: Checking Out Multiple Attachments of a Part

```
agileCheckOutAttachmentRequestType[0].setClassIdentifier("Part");
agileCheckOutAttachmentRequestType[0].setObjectNumber( partNumber );
CheckOutAttachmentType attachments[] = new CheckOutAttachmentType[1];
attachments[0] = new CheckOutAttachmentType();
attachments[0].setRowId(rowId);
int fileIds[] = new int[] {fileId};
attachments[0].setFiles(fileIds);
agileCheckOutAttachmentRequestType[0].setAttachments(attachments);
checkOutAttachmentRequestType.setRequests(agileCheckOutAttachmentRequestType);
```

---

**Note** See [Appendix](#) on page 281 for sample helper methods.

---

## Checking In the Attachments

You can Check-In a file attachment of an Agile object after it has undergone any modifications using the operation `checkInAttachment`. The attachment must be checked out prior to the 'check in' operation.

### Example: Checking In an Attachment to a Part

```
CheckInAttachmentRequestType checkInAttachmentRequestType = new
CheckInAttachmentRequestType();
AgileCheckInAttachmentRequestType agileCheckInAttachmentRequestType[] = new
AgileCheckInAttachmentRequestType[1];
agileCheckInAttachmentRequestType[0] = new AgileCheckInAttachmentRequestType();
agileCheckInAttachmentRequestType[0].setClassIdentifier("Part");
agileCheckInAttachmentRequestType[0].setObjectNumber(partNumber);
CheckInAttachmentType attachments[] = new CheckInAttachmentType[1];
attachments[0] = new CheckInAttachmentType();
```

```
attachments[0].setFileContent("Modified file information added after the  
checkin".getBytes() );  
attachments[0].setFileName("Modified_" + fileName);  
agileCheckInAttachmentRequestType[0].setAttachments(attachments);  
agileCheckInAttachmentRequestType[0].setRowId(rowId);  
checkInAttachmentRequestType.setRequest(agileCheckInAttachmentRequestType);
```

---

**Note** See sample [helper methods](#).

---

## Checking In an Attachment with FileId Identification

In the normal course of usage, rowId will prove to be sufficient in identifying the file. However, if the file that has been checked out is part of a row that contains multiple files, then fileId is essential to identify that particular file. For such cases, you have to use its fileId in conjunction with its rowId.

### Example: Checking In an Attachment using Field ID

```
CheckInAttachmentRequestType checkInAttachmentRequestType = new  
CheckInAttachmentRequestType();  
AgileCheckInAttachmentRequestType agileCheckInAttachmentRequestType[] = new  
AgileCheckInAttachmentRequestType[1];  
agileCheckInAttachmentRequestType[0] = new AgileCheckInAttachmentRequestType();  
gileCheckInAttachmentRequestType[0].setClassIdentifier("Part");  
agileCheckInAttachmentRequestType[0].setObjectNumber(partNumber);  
CheckInAttachmentType attachments[] = new CheckInAttachmentType[1];  
attachments[0] = new CheckInAttachmentType();  
attachments[0].setFileName("Modified_" + fileName[0]);  
attachments[0].setFileContent("Modified file information added after the  
checkin".getBytes() );  
attachments[0].setFileId(fileId);  
agileCheckInAttachmentRequestType[0].setAttachments(attachments);  
agileCheckInAttachmentRequestType[0].setRowId(rowId);  
checkInAttachmentRequestType.setRequest(agileCheckInAttachmentRequestType);
```

---

**Note** See sample [helper methods](#).

---

## Deleting the Attachments

Deleting an attachment is carried out using the operation removeRows.

See [Removing Rows from a Table](#) on page 44 for examples.

# Managing Workflows

**This chapter includes the following:**

---

▪ Getting the Status of a Workflow .....	62
▪ Getting the Workflow of a Routable Object.....	62
▪ Setting a Workflow .....	62
▪ Checking User Privileges.....	63
▪ Adding and Removing Approvers .....	63
▪ Getting Approvers .....	64
▪ Approving a Routable Object.....	64
▪ Rejecting a Routable Object .....	65
▪ Commenting a Change .....	66
▪ Auditing a Change .....	66
▪ Changing the Workflow Status of an Object.....	66

This chapter describes how to manage the Agile PLM workflows and provides sample code snippets.

## About Agile PLM Workflows

Agile has electronic routing, notification, and signoff capabilities, thus automating the change control process and providing a simplified but powerful workflow mechanism. With these workflow features, you can:

- Route changes automatically to the users who need to approve or observe the change.
- Send email alerts automatically to approvers and observers to notify them that a change has been routed to them.
- Approve or reject changes online.
- Attach comments to changes.

The workflow functionality available to each user for a particular routable object depends on the status of the routable object and the user's privileges. Your Agile API program should take these workflow dynamics into account and, where possible, adjust your program accordingly.

## How the Status of a Change Affects Workflow Functionality

The workflow actions available for a pending change are different from those for a released change. To check the status of a change to determine whether it's pending or released, use the operation `getStatus`. This operation returns an object for the workflow status.

## Getting the Status of a Workflow

The workflow actions available for a pending change are different from those for a released change. To check the status of a change to determine whether it's pending or released, use the operation `getStatus`. This operation returns **statusName** value for the workflow status, which are Pending, Submitted, Released, etc.

### Example: Getting the status of a change object

```
GetStatusRequestType getStatusRequestType = new GetStatusRequestType();
AgileGetStatusRequestType agileGetStatusRequestType[] = new AgileGetStatusRequestType[1];
agileGetStatusRequestType[0] = new AgileGetStatusRequestType();
agileGetStatusRequestType[0].setClassIdentifier("ECO");
agileGetStatusRequestType[0].setObjectNumber( changeNumber );
getStatusRequestType.setStatusRequest( agileGetStatusRequestType );
```

## Getting the Workflow of a Routable Object

When you create a new change, package, product service request, or quality change order, you must select a workflow. Otherwise, the object is in an unassigned state and cannot progress through a workflow process. Your Agile system can have multiple workflows defined for each type of routable object.

To get the valid workflows for a routable object, which has not yet been assigned a workflow, use the operation `getWorkFlows`.

### Example: Getting a Workflow

```
GetWorkflowsRequestType getWorkflowsRequestType = new GetWorkflowsRequestType();
AgileGetWorkflowsRequestType agileGetWorkflowsRequestType[] = new
AgileGetWorkflowsRequestType[1];
agileGetWorkflowsRequestType[0] = new AgileGetWorkflowsRequestType();
agileGetWorkflowsRequestType[0].setClassIdentifier("ECO");
agileGetWorkflowsRequestType[0].setObjectNumber( changeNumber );
getWorkflowsRequestType.setWorkflowRequest( agileGetWorkflowsRequestType );
```

## Setting a Workflow

If a change is still in the Pending state, you can deselect a workflow to make the change “unassigned” using the operation `setWorkFlow` and specifying the **setWorkFlowIdentifier** parameter.

As long as a change is in the Pending status, you can select a different workflow. Once a change moves beyond the Pending status, you can't change the workflow.

### Example: Setting a WorkFlow

```
SetWorkFlowRequestType setWorkFlowRequestType = new SetWorkFlowRequestType();
AgileSetWorkFlowRequestType agileSetWorkFlowRequestType[] = new
AgileSetWorkFlowRequestType[1];
agileSetWorkFlowRequestType[0] = new AgileSetWorkFlowRequestType();
agileSetWorkFlowRequestType[0].setClassIdentifier("ECO");
```

```

agileSetWorkFlowRequestType[0].setObjectNumber( changeNumber );
agileSetWorkFlowRequestType[0].setWorkFlowIdentifier( workflow );
setWorkFlowRequestType.setSetWorkFlowRequest( agileSetWorkFlowRequestType );

```

## Checking User Privileges

Agile privileges determine the types of workflow actions a user can perform on a Change Object. The Agile system administrator assigns roles and privileges to each user. Table below lists privileges needed to perform workflow actions.

Privilege	Related operation
Change Status	changeStatus
Comment	commentRObjct
Send	sendObject

To determine at run time whether a user has the appropriate privileges to perform a particular action, use the operation `checkPrivilege`. It yields a boolean value indicating if the user has the specified privilege.

Refer Schema documentation on Oracle eDelivery Site for `AgilePrivilegeType` enumerations.

### Example: Checking the privileges of a user

```

AgileUserIdentifierType user = new AgileUserIdentifierType();
user.setUserIdentifier("admin");

agileCheckPrivilegeRequestType[0].setUserIdentification(user);
AgilePrivilegeType privilege = AgilePrivilegeType.value1;
agileCheckPrivilegeRequestType[0].setPrivilege(privilege);
agileCheckPrivilegeRequestType[0].setClassIdentifier("Part");
agileCheckPrivilegeRequestType[0].setObjectNumber( partNumber );
checkPrivilegeRequestType.setRequests( agileCheckPrivilegeRequestType );

```

## Adding and Removing Approvers

After a change has been routed and the online approval process has begun, it may be necessary to add or remove people from the list of approvers or observers. To add or remove approvers or observers, a user must have both the Agile Product Change Server license and the Route privilege.

You don't need to load the Workflow table to modify the list of approvers. Once you have a routable object, such as an ECO, you can modify its list of approvers using the operation `addApprovers` and the operation `removeApprovers`. With these operations, you can specify the lists of approvers and observers, whether the notification is urgent, and an optional comment. The Agile Web Services provides overloaded operations for adding or removing a user or a user group from the list of approvers.

If any users you select as approvers or observers do not have appropriate privileges to view a

change, your program throws an [Exception](#) on page 20. To avoid the possible exception, check the privileges (checkPrivilege) of each user before adding him to the approvers or observers list.

**Example: Adding an Approver or Observer**

```
agileAddApproversRequestType[0].setClassIdentifier("ECO");
agileAddApproversRequestType[0].setObjectNumber( changeNumber );
agileAddApproversRequestType[0].setStatusIdentifier("CCB");
    AgileUserUserGroupIdentifierType users[] = new AgileUserUserGroupIdentifierType[2];
    users[0].setClassIdentifier("User");
    users[0].setObjectIdentifier(user1);
    users[1].setClassIdentifier(User);
    users[1].setObjectIdentifier(user2);
agileAddApproversRequestType[0].setApprovers(users);
agileAddApproversRequestType[0].setObservers(null);
agileAddApproversRequestType[0].setUrgent(false);
agileAddApproversRequestType[0].setComment("Comments");
addApproversRequestType.setAddApproversRequest(agileAddApproversRequestType);
```

**Example: Removing an Approver or Observer**

```
agileRemoveApproversRequestType[0].setClassIdentifier("ECO");
agileRemoveApproversRequestType[0].setObjectNumber( changeNumber );
agileRemoveApproversRequestType[0].setStatusIdentifier("CCB");
    AgileUserUserGroupIdentifierType usergroups[] = new
    AgileUserUserGroupIdentifierType[1];
    usergroups[0].setClassIdentifier("User");
    usergroups[0].setObjectIdentifier( USERNAME );
agileRemoveApproversRequestType[0].setApprovers(usergroups);
agileRemoveApproversRequestType[0].setObservers(null);
agileRemoveApproversRequestType[0].setComment("Comments");
removeApproversRequestType.setRemoveApproversRequest(agileRemoveApproversRequestType)
;
```

## Getting Approvers

Set the **statusIdentifier** in the operation getApprovers to obtain the list of approvers.

**Example: Getting Approvers for an Object**

```
agileGetApproversRequestType[0].setClassIdentifier("ECO");
agileGetApproversRequestType[0].setObjectNumber( changeNumber );
agileGetApproversRequestType[0].setStatusIdentifier( status );
getApproversRequestType.setApproversRequest(agileGetApproversRequestType);
```

## Approving a Routable Object

This method informs users the object is approved by the approver, or when the approver is approving the object on behalf of one or more user groups. You can also use this method to specify the secondSignature, escalations, transfers, or signoffForSelf parameters as they are set in server's

Preferences settings. Use the operation approveRObjct.

**Example: Approving a Routable Object and notifying the users**

```
AgileUserUserGroupIdentifierType notifiers[] = new AgileUserUserGroupIdentifierType[1];
notifiers[0] = new AgileUserUserGroupIdentifierType();
notifiers[0].setClassIdentifier("User");
notifiers[0].setObjectIdentifier( notifier1 );
AgileApproveObjectRequestType agileApproveObjectRequestType[] = new
AgileApproveObjectRequestType[1];
agileApproveObjectRequestType[0] = new AgileApproveObjectRequestType();
    agileApproveObjectRequestType[0].setClassIdentifier("ECO");
    agileApproveObjectRequestType[0].setObjectNumber( changeNumber );
    agileApproveObjectRequestType[0].setPassword( PASSWORD );
    agileApproveObjectRequestType[0].setComment("Comment");
    agileApproveObjectRequestType[0].setSecondSignature(null);
    agileApproveObjectRequestType[0].setNotifiers(notifiers);
    agileApproveObjectRequestType[0].setEscalations(null);
    agileApproveObjectRequestType[0].setTransfers(null);
    agileApproveObjectRequestType[0].setApproveForGroup(null);
    agileApproveObjectRequestType[0].setSignoffForSelf(true);
approveObjectRequestType.setApproveObject( agileApproveObjectRequestType );
```

## Rejecting a Routable Object

This method informs users that the routable object is rejected by the approver, or when the approver is rejecting the object on behalf of one or more user groups. You can also use this method to specify the secondSignature, escalations, transfers, or SignoffForSelf parameters as they are set in server's Preferences settings. Use the operation rejectRObjct.

**Example: Rejecting a Routable Object and notifying the users**

```
agileRejectObjectRequestType[0].setClassIdentifier("ECO");
agileRejectObjectRequestType[0].setObjectNumber( changeNumber );
agileRejectObjectRequestType[0].setPassword( PASSWORD );
agileRejectObjectRequestType[0].setComment("Comment");
agileRejectObjectRequestType[0].setSecondSignature(null);
    AgileUserUserGroupIdentifierType notifiers[] = new
    AgileUserUserGroupIdentifierType[1];
    notifiers[0] = new AgileUserUserGroupIdentifierType();
    notifiers[0].setClassIdentifier("User");
    notifiers[0].setObjectIdentifier( notifier1 );
    agileRejectObjectRequestType[0].setNotifiers(notifiers);
    agileRejectObjectRequestType[0].setEscalations(null);
    agileRejectObjectRequestType[0].setTransfers(null);
    agileRejectObjectRequestType[0].setRejectForGroups(null);
    agileRejectObjectRequestType[0].setSignoffForSelf(true);
rejectObjectRequestType.setRejectObject( agileRejectObjectRequestType );
```

## Commenting a Change

Use the operation `CommentRObj` operation to comment a change. Use boolean variables to denote whether the originators, change analysts and CCB need to be notified.

```
agileCommentRObjRequestType[0].setClassIdentifier("ECO");
agileCommentRObjRequestType[0].setObjectNumber( changeNumber );
agileCommentRObjRequestType[0].setComment("Comment");
agileCommentRObjRequestType[0].setNotifyOriginator(true);
agileCommentRObjRequestType[0].setNotifyChangeAnalyst(true);
agileCommentRObjRequestType[0].setNotifyCCB(false);
AgileUserUserGroupIdentifierType notifyList[] = new AgileUserUserGroupIdentifierType[1];
notifyList[0] = new AgileUserUserGroupIdentifierType();
notifyList[0].setClassIdentifier("User");
notifyList[0].setObjectIdentifier( USERNAME );
agileCommentRObjRequestType[0].setNotifyList(notifyList);
commentRObjRequestType.setCommentRObjRequest( agileCommentRObjRequestType );
```

## Auditing a Change

Auditing a routable object, like an ECO, requires specifying the type of routable object in the operation `AuditRObj`.

```
agileAuditRObjRequestType[0].setClassIdentifier("ECO");
agileAuditRObjRequestType[0].setObjectNumber( changeNumber );
agileAuditRObjRequestType[0].setAuditRelease(true);
auditRObjRequestType.setRequest( agileAuditRObjRequestType );
```

## Changing the Workflow Status of an Object

Use the operation `changeStatus` to change the workflow status of an object.

```
agileChangeStatusRequestType[0].setClassIdentifier("ECO");
agileChangeStatusRequestType[0].setObjectNumber( changeNumber );
agileChangeStatusRequestType[0].setNewStatusIdentifier( newStatus );
AgileUserUserGroupIdentifierType users[] = new AgileUserUserGroupIdentifierType[1];
users[0] = new AgileUserUserGroupIdentifierType();
users[0].setClassIdentifier("User");
users[0].setObjectIdentifier( user1 );
agileChangeStatusRequestType[0].setApprovers(users);
agileChangeStatusRequestType[0].setObservers(null);
agileChangeStatusRequestType[0].setNotifiers(null);
agileChangeStatusRequestType[0].setComment("Comments");
agileChangeStatusRequestType[0].setPassword("password");
agileChangeStatusRequestType[0].setAuditRelease(false);
agileChangeStatusRequestType[0].setUrgent(false);
```



```
agileChangeStatusRequestType[0].setNotifyOriginator(true);  
agileChangeStatusRequestType[0].setNotifyChangeAnalyst(true);  
agileChangeStatusRequestType[0].setNotifyCCB(true);  
changeStatusRequestType.setChangeStatusRequest(agileChangeStatusRequestType);
```



# Creating and Loading Queries

This chapter includes the following:

▪ Creating a Query .....	69
▪ Creating a New Query from an Existing Query Object .....	69
▪ Loading a Query .....	70
▪ Executing a Saved Query .....	70
▪ Updating a Saved Query .....	70
▪ Deleting a Query .....	71
▪ Creating and Loading Queries .....	71

An Query is an object that defines how to search for Agile PLM data. It defines a search similar to the searches that you can use in Agile Web Client. The search can have multiple search criteria (like an Advanced Search in Agile Web Client), or it can be a simple search that specifies only one criterion.

## Creating a Query

To create and execute a query, you must first create an `Query` object. As with other Agile API objects, you create the object using the `createQuery` operation.

In its simplest form, the parameters that you pass with the `createQuery` operation to create a query are the `Query` object type and the query class used in the search.

### Example: Creating a query

```
CreateQueryRequestType createQueryRequestType =
    new CreateQueryRequestType();
AgileCreateQueryRequestType[] agileCreateQueryRequestType =
    new AgileCreateQueryRequestType[1];
agileCreateQueryRequestType[0] = new AgileCreateQueryRequestType();
QueryObjectIdentifierType queryObjectIdentifierType = new QueryObjectIdentifierType();
queryObjectIdentifierType.setName("Personal Searches/Submitted Changes");
QueryObjectType queryObjectType = new QueryObjectType();
queryObjectType.setOwner("badriv");
agileCreateQueryRequestType[0].setQueryParams(queryObjectType);
createQueryRequestType.setRequests(agileCreateQueryRequestType);
```

The query class you specify with the `createQuery` operation also includes objects from all of its subclasses. For example, if you search for objects in the `Item` class, the results include parts and documents. If you search for objects in the `Change` class, the results include objects from all `Change` subclasses (`Deviation`, `ECO`, `ECR`, `MCO`, `PCO`, `SCO`, and `Stop Ship`). If you want to search only a specific subclass, you should explicitly specify that class.

## Creating a New Query from an Existing Query Object

You can also use the `saveAsQuery` operation to create a new query from an existing query object.

**Example: Using IQuery.saveAs() to save a query to a folder**

```
SaveAsQueryRequestType saveAsQueryRequestType =
    new SaveAsQueryRequestType();
AgileSaveAsQueryRequestType[] agileSaveAsQueryRequestType =
    new AgileSaveAsQueryRequestType[1];
agileSaveAsQueryRequestType[0] = new AgileSaveAsQueryRequestType();
QueryObjectIdentifierType queryObjectIdentifierType = new QueryObjectIdentifierType();
queryObjectIdentifierType.setName("Personal Searches/Submitted Changes");
agileSaveAsQueryRequestType[0].setQueryIdentifier(queryObjectIdentifierType);
agileSaveAsQueryRequestType[0].setNewQueryIdentifier("Personal Searches/My Submitted
Changes");
saveAsQueryRequestType.setRequests(agileSaveAsQueryRequestType);
```

## Loading a Query

**Example: Loading a query**

```
LoadQueryRequestType loadQueryRequestType =
    new LoadQueryRequestType();
AgileLoadQueryRequestType[] agileLoadQueryRequestType =
    new AgileLoadQueryRequestType[1];
agileLoadQueryRequestType[0] = new AgileLoadQueryRequestType();
QueryObjectIdentifierType queryObjectIdentifierType = new QueryObjectIdentifierType();
queryObjectIdentifierType.setName("Recycle Bin Searches/Deleted Changes");
agileLoadQueryRequestType[0].setQueryIdentifier(queryObjectIdentifierType);
loadQueryRequestType.setRequests(agileLoadQueryRequestType);
```

## Executing a Saved Query

**Example: Executing an existing Query saved in the Agile system**

```
ExecuteSavedQueryRequestType executeSavedQueryRequestType =
    new ExecuteSavedQueryRequestType();
AgileExecuteSavedQueryRequestType[] agileExecuteSavedQueryRequestType =
    new AgileExecuteSavedQueryRequestType[1];
agileExecuteSavedQueryRequestType[0] = new AgileExecuteSavedQueryRequestType();
QueryObjectIdentifierType queryObjectIdentifierType = new QueryObjectIdentifierType();
queryObjectIdentifierType.setName("Recycle Bin Searches/Deleted Changes");
agileExecuteSavedQueryRequestType[0].setQueryIdentifier(queryObjectIdentifierType);
executeSavedQueryRequestType.setRequests(agileExecuteSavedQueryRequestType);
```

## Updating a Saved Query

**Example: Updating an existing Query**

```
UpdateQueryRequestType updateQueryRequestType =
    new UpdateQueryRequestType();
AgileUpdateQueryRequestType[] agileUpdateQueryRequestType =
    new AgileUpdateQueryRequestType[1];
agileUpdateQueryRequestType[0] = new AgileUpdateQueryRequestType();
QueryObjectIdentifierType queryObjectIdentifierType = new QueryObjectIdentifierType();
queryObjectIdentifierType.setName("Personal Searches/Submitted Changes");
agileUpdateQueryRequestType[0].setQueryIdentifier(queryObjectIdentifierType);
QueryObjectType queryObjectType = new QueryObjectType();
```

```
queryObjectType.setOwner("badriv");
agileUpdateQueryRequestType[0].setNewQueryParams(queryObjectType);
updateQueryRequestType.setRequests(agileUpdateQueryRequestType);
```

## Deleting a Query

To delete a query that has been saved, use the `deleteQuery` operation.

Temporary queries, that is, queries that are created but not saved to a folder are automatically deleted after the user session is closed. For lengthy sessions, you can use the `deleteQuery` operation to explicitly delete a temporary query after you're finished running it.

### Example: Deleting a Query

```
DeleteQueryRequestType deleteQueryRequestType =
    new DeleteQueryRequestType();
AgileDeleteQueryRequestType[] agileDeleteQueryRequestType =
    new AgileDeleteQueryRequestType[1];
agileDeleteQueryRequestType[0] = new AgileDeleteQueryRequestType();
QueryObjectIdentifierType queryObjectIdentifierType = new
QueryObjectIdentifierType();
queryObjectIdentifierType.setName("Personal
Searches/MySubmittedChanges");
agileDeleteQueryRequestType[0].setQueryIdentifier(queryObjectIdentifier
Type);
deleteQueryRequestType.setRequests(agileDeleteQueryRequestType);
```

## Working with Searches

This section describes how to work with the Agile PLM Searches and provides sample code snippets.

## Agile PLM Searches

Agile PLM Searches can have multiple search criteria (like an Advanced Search in the Agile Web Client), or it can be a simple search that specifies only one criteria.

### Specifying Search Criteria

You can narrow the number of objects returned from a search by specifying search criteria. If you specify `*` as the search criteria, the query returns references to all objects in the specified query class. It's a good practice to limit the search criteria as much as possible, as the amount of data returned may be excessively large, resulting in decreased performance.

You can use the `setCriteria(criteria)` method to specify query criteria, which sets the search criteria from a specified `String`. This `String` references one or more parameters.

```
advancedSearchRequestType.setClassIdentifier("Part");
advancedSearchRequestType.setCaseSensitive(false);
String criteria = "[Title Block.Number] contains 'PO' && " +
```

```
"[Title Block.Description] is not null";
advancedSearchRequestType.setCriteria(criteria);
String attribute1 = "Title Block.Number";
String attribute2 = "Title Block.Description";
String attribute3 = "Title Block.Lifecycle Phase";
advancedSearchRequestType.setResultAttributes(new String[]{attribute1,
attribute2, attribute3});
advancedSearchRequestType.setDisplayName("Search123");
AdvancedSearchResponseType advancedSearchResponseType =
agileStub.advancedSearch(advancedSearchRequestType);
```

## Search Conditions

The Agile Web Services provides a simple yet powerful query language for specifying search criteria. The query language defines the proper syntax for filters, conditions, attribute references, relational operators, logical operators, and other elements.

Search criteria consist of one or more search conditions. Each search condition contains the following elements:

- **Left operand** – The left operand is always an attribute enclosed in brackets, such as `[Title Block.Number]`. You can specify the attribute as an attribute name (fully qualified name or short name) or attribute ID number. The attribute specifies which characteristic of the object to use in the search.
- **Relational operator** – The relational operator defines the relationship that the attribute has to the specified value, for example, “equal to” or “not equal to.”
- **Right operand** – The matching value for the specified attribute in the left operand. The right operand can be a constant expression or a set of constant expressions. A set of constant expressions is needed if the relational operator is “between,” “not between,” “in,” or “not in.”

Following is an example of a search condition:

```
[Title Block.Description] == 'Computer'
```

This is another example where the right operand is a set of constant expressions:

```
[Page Two.Numeric01] between ('1000', '2000')
```

## Search Operation Keywords

When you specify a search condition, you must use proper keywords to construct the statement. The following keywords are available:

and	does	less	or	to
asc	equal	like	order	union
between	from	minus	phrase	where
by	greater	none	select	with
contain	in	not	start	word

contains      intersect      null      starts      words  
 desc      is      of      than

These keywords are not localized. You must use English keywords, regardless of locale. You can use the keywords in lower case or upper case. In addition to keywords, you can use Agile PLM variables such as `$USER` (for current user) and `$TODAY` (for today's date) in Agile Searches.

The "in" operator does not support MultiList in (set) query criteria.

## Specifying Search Attributes

Every Agile PLM object that you can search for also has an associated set of attributes, which are inherent characteristics of the object. You can use these attributes as the left operand of a search condition. The right operand of the search condition specifies the attribute's value(s).

A search attribute must be enclosed within brackets, for example, `[Title Block.Number]`. The brackets are needed because many attribute names have spaces. If a search attribute is not enclosed within brackets, your query will fail.

You can specify a search attribute in the following ways:

Attribute reference	Example
attribute ID number	<code>[1001]</code>
fully-qualified attribute name	<code>[Title Block.Number]</code>
short attribute name	<code>[Number]</code>

**Note** Because attribute names can be modified, Agile recommends referencing attributes by ID number or constant. However, many of the examples in this chapter reference attributes by name simply to make them more readable. If you choose to reference attributes by name, use the fully-qualified attribute name instead of the short name. Short attribute names are not guaranteed to be unique and could therefore cause your query to fail or produce unexpected results.

**Note** Specifying the search attributes using Attribute APIName is not supported.

Attribute names, whether you use the long or short form, are case-insensitive. For example, `[Title Block.Number]` and `[TITLE BLOCK.NUMBER]` are both allowed. Attribute names are also localized. The names of Agile PLM attributes vary based on the locale of your Agile Application Server. If you are creating a query that is going to be used on servers in different locales, you should reference attributes by ID number (or the equivalent constant) instead of by name.

If the attribute name contains special characters, such as quotes or backslashes, you can type these characters using the backslash (`\`) as an escape character. For example, to include a quote character in your string, type `\'`. If you want to write a backslash, type two of them together (`\\`). If the attribute name contains square brackets, enclose the entire name in quotes:

```
['Page Two.Unit of Measure [g or oz]']
```

There are other, perhaps less intuitive, ways to specify attributes. For example, you could pass in an `IAttribute` reference using a parameter of the `setCriteria()` method. In the following example, `'%0'` references the attribute in the `params` parameter.

```
advancedSearchRequestType.setCriteria("[Title Block.Number] starts with %0 and [Title  
Block.Part Category] in %1 and [Title Block.Description] contains %2");  
ParamListType[] params = new ParamListType[3];  
params[0] = new ParamListType();  
params[0].setParameter(new String[]{"P00"});  
params[1] = new ParamListType();  
params[1].setParameter(new String[]{"Electrical", "Mechanical"});  
params[2] = new ParamListType();  
params[2].setParameter(new String[]{"Resistor"});  
advancedSearchRequestType.setParams(params);
```

You can also use `String` concatenation to reference an attribute constant:

```
advancedSearchRequestType.setCriteria("[\" + ItemConstants.ATT_TITLE_BLOCK_DESCRIPTION +  
\"] == 'Computer'");
```

## Getting the Searchable Attributes

The searchable attributes for a query depend on the specified query class or subclass. However, the searchable attributes for a subclass can differ greatly from searchable attributes for its parent class.

Due to database considerations, not all attributes are searchable. Generally, a few select Page One attribute (namely: Title Page, Cover Page, and General Info attributes) are searchable for each class.

If a tab is not configured in Java Client to be visible, you can still search for an attribute on that tab in the Agile Web Services. However, you must search for the Table name that corresponds to the Tab name.

To find the searchable attributes for a query, use the `getSearchableAttributes` operation.

Even though an attribute may not be searchable, it can still be included as a column in the query results. For more information, see [Setting Result Attributes for a Query](#).

## Using Relational Operators

Table below lists relational operators that are supported by the Agile Web Services search operations.

English operator	Notation	Description
equal to	==	Finds only an exact match with the specified value.
not equal to	!=	Finds any value other than an exact match with the specified value.
greater than	>	Finds any value greater than the specified value.
greater than or equal to	>=	Finds any value greater than or equal to the specified value.
less than	<	Finds any value less than the specified value.
less than or equal to	<=	Finds any value less than or equal to the specified value.
contains, contains all		Finds any value that includes the specified value.



English operator	Notation	Description
does not contain, does not contain all		Finds any value that does not include the specified value.
contains any		Finds any value that includes the specified value.
does not contain any		Finds any value that does not include the specified value.
contains none of		Finds any value that includes none of the specified values.
does not contain none of		Behaves the same as <code>does not contain any</code> .
starts with		Finds values that begin with characters in the specified value.
does not start with		Finds values that do not begin with characters in the specified value.
is null		Finds objects where the selected attribute contains no value.
is not null		Finds objects where the selected attribute contains a value.
like		Performs a wildcard search, finding objects that match a single character or any string.
not like		Performs a wildcard search, finding objects that do not match a single character or any string.
between		Finds objects that fall between the specified values.
not between		Finds objects that do not fall between the specified values.
in		Finds objects that match any of the specified values.
not in		Finds objects that do not match any of the specified values.
contains phrase		Finds objects with files that contain the specified phrase.
contains all words		Finds objects with files that contain all of the specified words.
contains any word		Finds objects with files that contain any of the specified words.
contains none of		Finds objects with files that contain none of the specified words.

Relational operators are not localized. You must use English keywords, regardless of locale. As with other query language keywords, you can use them in lower case or upper case.

### Using Unicode Escape Sequences

Agile Web Services Search operations support Unicode escape sequences. The primary usage of Unicode escape sequences in a query string is to search for nonburnable or foreign local character sets. A Unicode character is represented with the Unicode escape sequence `\uxxxx`, where `xxxx` is a sequence of four hexadecimal digits.

For example, to search for an item with Unicode 3458, use the following query:

```
Select * from [Items] where [Description] contains '\u3458'
```

There is another query operation for “contains” usage in the case of MultiList.

### Using Between, Not Between, In, and Not In Operators

The 'between', 'not between', 'in', and 'not in' relational operators are not supported directly by Agile PLM Java and Web clients. These relational operators provide a convenient shorthand method for specifying 'equal to', 'not equal to', 'greater than or equal to', or 'less than or equal to' operations with a set of values.

Short form	Equivalent long form
[Number] between ('1','6')	[Number] >= '1' and [Number] <= '6'
[Number] not between ('1','6')	[Number] < '1' and [Number] > '6'
[Number] in ('1','2','3','4','5','6')	[Number] == '1' or [Number] == '2' or [Number] == '3' or [Number] == '4' or [Number] == '5' or [Number] == '6'
[Number] not in ('1','2','3','4','5','6')	[Number] != '1' and [Number] != '2' and [Number] != '3' and [Number] != '4' and [Number] != '5' and [Number] != '6'

As shown in the table, when you use the 'between', 'not between', 'in', and 'not in' relational operators, each value in the set of values must be enclosed in quotes and delimited by commas. Here are more criteria examples that use 'between' and 'in' relational operators:

```
[Title Block.Number] in ('1000-02', '1234-01', '4567-89')
[Title Block.Effectivity Date] between ('01/01/2001', '01/01/2002')
[Page Two.Numeric01] between ('1000', '2000')
```

**Note** The relational operators any, all, none of, and not all are not supported in the Web Services.

### Using the Nested Criteria to Search for Values in Object Lists

Several lists in Agile PLM contain business objects, such as Agile PLM users. To search for an object in such a list, you can specify nested query criteria. Nested criteria are enclosed in parentheses and separated from each other by a logical AND (&&) or OR (||) operator. A comma can also be used to separate nested criteria; it's equivalent to a logical OR.

The following criteria finds a user with the first name Christopher OR the last name Nolan.

```
[Page Two.Create User] in ([General Info.First Name] == 'Christopher',
[General Info.Last Name] == 'Nolan')
```

The following criteria finds a user with the first name Christopher AND the last name Nolan.

```
[Page Two.Create User] in ([General Info.First Name] == 'Christopher' &&
[General Info.Last Name] == 'Nolan')
```

The parameter query is not supported in nested queries and multiple values for one placeholder in query parameters must be specified in two dimensional arrays as shown in the example below.

#### Example: Correct and incorrect parameter query in nested query criteria

- The parameter query specified in the following nested query criteria will fail to execute:

```
[Page Two.User1] in ([General Info.First Name] == %0)
```
- However, when it is explicitly specified as a string value, instead of the placeholder, it will succeed:

```
[Page Two.User1] in ([General Info.First Name] == 'Christopher')
```

### Searching for Words or Phrases Contained in Attachments

Two special attributes, [Attachments.File Document Text] and [Files.Document Text], are used to index the content of files stored on the Agile file management server. If you are hosting your database on Oracle, you can take advantage of a feature that lets you search for words or phrases contained in attachments. When you create search criteria that uses either of these attributes, there are four additional relational operators you can use:

- contains phrase
- contains all words
- contains any word
- contains none of

The following table shows several search conditions that search for words or phrases in attachments.

Search Condition	Finds
[Attachments.File Document Text] <b>contains phrase</b> 'adding new materials'	Objects in which any of their attachments contain the phrase "adding new materials."
all [Attachments.File Document Text] <b>contains all words</b> 'adding new materials'	Objects in which all their attachments contain the words "adding," "new," and "materials."
none of [Attachments.File Document Text] <b>contains any word</b> 'containers BOM return output'	Objects in which none of their attachments contain any of the words "containers," "BOM," "return," or "output."
[Attachments.File Document Text] <b>contains none of</b> 'containers BOM output'	Objects in which any of their attachments do not contain the words "containers," "BOM," or "output."

### Using Logical Operators

You can use logical operators to combine multiple search conditions into a complex filter. When you have two or more conditions defined in a set of query criteria, the relationship between them is defined as either 'and' or 'or'.

- **and** narrows the search by requiring that both conditions are met. Each item in the results must match both conditions. The 'and' logical operator can also be specified using two ampersands, '&&'.
- **or** broadens the search by including any object that meets either condition. Each item in the results table needs to match only one of the conditions, but may match both. The 'or' logical operator can also be specified using two vertical bars, '||'.

Logical operators are case-insensitive. For example, 'and' or 'AND' are both allowed.

The following query criteria finds parts that have both a part category equal to Electrical and a

lifecycle phase equal to Inactive.

```
[Title Block.Part Category] == 'Electrical' and  
[Title Block.Lifecycle Phase] == 'Inactive'
```

If you replace the ‘and’ operator with ‘or’, the query locates all parts with either a part category of Electrical or a lifecycle phase of Inactive, which could be a large number of parts.

```
[Title Block.Part Category] == 'Electrical' or  
[Title Block.Lifecycle Phase] == 'Inactive'
```

The Agile Web Services provides three where-used set operators. For more information, see [Creating a Where-Used Query](#).

Logical operators, including the where-used set operators, are not localized. You must use English keywords, regardless of locale.

## Using Wildcard Characters with the Like Operator

If you define a search condition using the ‘like’ operator, you can use two wildcard characters: the asterisk (\*) and question mark (?). The asterisk matches any string of any length, so **\*at** finds cat, splat, and big hat.

For example, `[Title Block.Description] like '*book*'` returns all objects that contain the word “book,” such as textbook, bookstore, books, and so on.

The question mark matches any single character, so **?at** finds hat, cat, and fat, but not splat.

For example, `[Title Block.Description] like '?al*'` matches any word containing “al” that is preceded by a single letter, such as tall, wall, mall, calendar, and so on.

## Using Parentheses in Search Criteria

Where-used, set operators have higher priority than **and** and **or** logical operators, as shown by the following table.

Priority	Operator(s)
1	<ul style="list-style-type: none"><li>□ union</li><li>□ intersection</li><li>□ minus</li></ul>
2	<ul style="list-style-type: none"><li>□ and</li><li>□ or</li></ul>

Therefore, search conditions joined by **union**, **intersection**, and **minus** operators are evaluated before conditions joined by **and** or **or**.

If you use where-used set operators (‘union’, ‘intersect’, or ‘minus’) in search criteria, you can use parentheses to change the order that criteria are evaluated. If only ‘and’ or ‘or’ logical operators are used in a search criteria, additional parentheses aren’t needed because they don’t change the result of criteria evaluation.

The following two criteria, although they contain the same search conditions, provide different results because parentheses are placed differently:

```
([Title Block.Part Category] == 'Electrical' and
```

```

[Title Block.Description] contains 'Resistor') union
([Title Block.Description] contains '400' and
[Title Block.Product Line(s)] contains 'Taurus')

[Title Block.Part Category] == 'Electrical' and
([Title Block.Description] contains 'Resistor' union
[Title Block.Description] contains '400') and
[Title Block.Product Line(s)] contains 'Taurus'

```

## Using SQL Syntax to specify Search Criteria

In addition to its standard query language, the Agile Web Services also supports SQL-like syntax to specify search criteria. If you are familiar with SQL statements, you may find this extended query language more flexible, more powerful and easier to work with. It combines in one operation the specification of the query result attributes, the query class, the search condition, and the sort column(s).

This is a simple example that demonstrates the syntax:

- Query result attributes: `SELECT [Title Block.Number], [Title Block.Description]`
- Query class: `FROM [Items]`
- Search condition: `WHERE [Title Block.Number] starts with 'P'`
- Sort column(s): `ORDER BY 1 asc`

To improve readability, it's recommended that SQL key words such as `SELECT` and `FROM` are all typed using capital letters and each part of the statement appears on a separate line. This is merely a convention, not a requirement. SQL key words are not case-sensitive, and you can write the entire query string on one line if you prefer.

The best way to demonstrate the advantages of SQL syntax is to compare the code for a query that uses standard Agile API query syntax for search criteria with one that uses SQL syntax. The following example shows a query created using the standard Agile API query syntax:

### Example: Query using standard Agile API query syntax

```

advancedSearchRequestType.setCriteria("[Page Two.Numeric01] between (1000, 2000)");
//Set result attributes
String[] attrs = { "Title Block.Number", "Title Block.Description",
    "Title Block.Lifecycle Phase" };
advancedSearchRequestType.setResultAttributes(attrs);

```

This example shows the same query rewritten in SQL syntax. Although the example doesn't have fewer lines of code, you may find that it's more readable than Agile API query syntax, particularly if you're familiar with SQL.

### Example: Search criteria using SQL syntax

```

String criteria = "SELECT " +
    "[Title Block.Number],[Title Block.Description], " +
    "[Title Block.Lifecycle Phase] " +
    "FROM " +
    "[Items] " +
    "WHERE " +
    "[Title Block.Number] between (1000, 2000)";

```

```
advancedSearchRequestType.setCriteria(criteria);
```

The following example shows a query written with SQL syntax that specifies the search criteria.

**Example: Using SQL syntax to specify query attributes**

```
try {
    String statement =
        "SELECT " +
        "[Title Block.Number], [Title Block.Description] " +
        "FROM " +
        "[Items] " +
        "WHERE " +
        "[Title Block.Description] like %0";
    advancedSearchRequestType.setCriteria(statement);
}
```

---

**Note** Remember, the FROM part of the search condition specifies the query class. If you use the classIdentifier attribute to also specify a query class, the query class specified in the SQL search condition takes precedence.

---

## Using SQL Wildcards

You can use both the asterisk (\*) and question mark (?) as wildcards in a query that uses SQL syntax. In Agile Web Services search operation. The asterisk matches any string and the question mark matches any single character. You can use wildcards in the SELECT statement (the specified query result attributes) and the WHERE statement (the search condition).

For example, "SELECT \*" specifies all available query result attributes.

## Setting Result Attributes for a Search

When you use the operation advancedSearch, it returns several output fields, which are also called result attributes. By default, there are only a few result attributes for each query class. You can add or remove result attributes using the `setResultAttributes()`.

The following table shows the default query result attributes for each predefined Agile PLM class.

Query class	Default result attributes
<b>Changes</b> Change Orders ECO Change Requests ECR Deviations Deviation Manufacturer Orders MCO Price Change Orders PCO Sites Change Orders SCO Stop Ships Stop Ship	Cover Page.Change Type Cover Page.Number Cover Page.Description Cover Page.Status Cover Page.Workflow
<b>Customers</b> Customers Customer	General Info.Customer Type General Info.Customer Number General Info.Customer Name General Info.Description General Info.Lifecycle Phase
<b>Declarations</b> Homogeneous Material Declarations Homogeneous Material Declaration IPC 1752-1 Declarations IPC 1752-1 Declaration IPC 1752-2 Declarations IPC 1752-2 Declaration JGPSSI Declarations JGPSSI Declaration Part Declarations Part Declaration Substance Declarations Substance Declaration Supplier Declarations of Conformance Supplier Declaration of Conformance	Cover Page.Name Cover Page.Description Cover Page.Supplier Cover Page.Status Cover Page.Workflow Cover Page.Compliance Manager Cover Page.Due Date Cover Page.Declaration Type

Query class	Default result attributes
<b>Discussions</b> Discussions Discussion	Cover Page.Subject Cover Page.Status Cover Page.Priority Cover Page.Type
<b>File Folders</b> File Folders File Folder	Title Block.Type Title Block.Number Title Block.Description Title Block.Lifecycle Phase
<b>Items</b> Parts Part Documentation Document	Title Block.Item Type Title Block.Number Title Block.Description Title Block.Lifecycle Phase Title Block.Rev
<b>Manufacturers</b> Manufacturers Manufacturer	General Info.Name General Info.City General Info.State General Info.Lifecycle Phase General Info.URL
<b>Manufacturer Parts</b> Manufacturer Parts Manufacturer Part	General Info.Manufacturer Part Number General Info.Manufacturer Name General Info.Description General Info.Lifecycle Phase
<b>Packages</b> Packages Package	Cover Page.Package Number Cover Page.Description Cover Page.Assembly Number Cover Page.Status Cover Page.Workflow
<b>Part Groups</b> Part Groups Commodity Part Family	General Info.Name General Info.Description General Info.Lifecycle Phase General Info.Commodity Type General Info.Overall Compliance



Query class	Default result attributes
<b>Prices</b> Published Prices Contracts Published Price Quote History Quote History	General Info.Price Number General Info.Description General Info.Rev General Info.Price Type General Info.Lifecycle Phase General Info.Program General Info.Customer General Info.Supplier
<b>Product Service Requests</b> Non-Conformance Reports NCR Problem Reports Problem Report	Cover Page.PSR Type Cover Page.Number Cover Page.Description Cover Page.Status Cover Page.Workflow
<b>Programs</b> Activities Program Phase Task Gates Gate	General Info.Name General Info.Description General Info.Status General Info.Health General Info.Owner General Info.Root Parent General Info.Workflow General Info.Type
<b>Projects</b> Sourcing Projects Sourcing Project	General Info.Project Type General Info.Number General Info.Description General Info.Manufacturing Site General Info.Ship To Location General Info.Program General Info.Customer General Info.Lifecycle Phase
<b>Quality Change Requests</b> Corrective Action/Preventive Action CAPA Audits Audit	Cover Page.QCR Type Cover Page.QCR Number Cover Page.Description Cover Page.Status

Query class	Default result attributes
	Cover Page.Workflow
<b>RFQ Responses</b> RFQ Responses RFQ Response	Cover Page.RFQ Number Cover Page.RFQ Description Cover Page.Lifecycle Phase Cover Page.Requested Cover Page.Completed Cover Page.Due Date
<b>RFQs</b> RFQs RFQ	Cover Page.RFQ Number Cover Page.RFQ Description Cover Page.MFG Site Cover Page.Ship-To Location Cover Page.Program Cover Page.Customer Cover Page.Lifecycle Phase Cover Page.RFQ Type
<b>Sites</b> Sites Site	General Info.Name General Info.Contact General Info.Phone
<b>Specifications</b> Specifications Specification	General Info.Name General Info.Description General Info.Lifecycle Phase General Info.Jurisdictions General Info.Validation Type General Info.Specification Type

Query class	Default result attributes
<b>Substances</b> Materials Material Subparts Subpart Substance Groups Substance Group Substances Substance	General Info.Name General Info.Description General Info.CAS Number General Info.Lifecycle Phase General Info.Substance Type
<b>Suppliers</b> Suppliers Component Manufacturer Contract Manufacturer Distributor Manufacturer Rep	General Info.Supplier Type General Info.Number General Info.Name General Info.Description General Info.Status
<b>Transfer Orders</b> Content Transfer Orders CTO Automated Transfer Orders ATO	Cover Page.Transfer Order Type Cover Page.Transfer Order Number Cover Page.Description Cover Page.Status Cover Page.Workflow

## Specifying Result Attributes

If you run a query and find that the resulting `table` object does not contain the attributes you expected, it's because you didn't specify result attributes. The following example shows how to specify the result attributes for a query.

### Example: Setting query result attributes

```
String attribute1 = "Title Block.Number";
String attribute2 = "Title Block.Description";
String attribute3 = "Title Block.Lifecycle Phase";
advancedSearchRequestType.setResultAttributes(new String[]{attribute1, attribute2,
attribute3} );
```

The `ResultAttributes` element takes an array of `String` where you can array of attribute names (such as {"Title Block.Description", "Title Block.Number"}) or attribute ID constants or attribute API names. The following example shows how to specify result attributes using ID constants.

### Example: Setting query result attributes by specifying ID constants

```
private void setQueryResultColumns(IQuery query) throws APIException {
    // Put the attribute IDs into an array
    String[] attrs = { ItemConstants.ATT_TITLE_BLOCK_NUMBER+""",
                      ItemConstants.ATT_TITLE_BLOCK_DESCRIPTION+""",
                      ItemConstants.ATT_TITLE_BLOCK_LIFECYCLE_PHASE+""",
                      ItemConstants.ATT_PAGE_TWO_TEXT01+""",
```

```
        ItemConstants.ATT_PAGE_TWO_NUMERIC01+""",
        ItemConstants.ATT_PAGE_THREE_TEXT01+"""};
// Set the result attributes for the query
advancedSearchRequestType.setResultAttributes(attrs);
}
```

When you use the `setResultAttributes()` method, make sure you specify valid result attributes. Otherwise, the `setResultAttributes()` method will fail.

## Examples of Searches

The examples below show how to create quick search, advanced search and how to get the searchable attributes.

### Quick Search

Operation - quickSearch

```
QuickSearchRequestType quickSearchRequestType = new QuickSearchRequestType();
quickSearchRequestType.setClassIdentifier("Part");
quickSearchRequestType.setKeywords("P0*");
quickSearchRequestType.setSearchFiles(false);
QuickSearchResponseType quickSearchResponseType =
    agileStub.quickSearch(quickSearchRequestType);
```

### Advanced Search

Operation - advancedSearch

```
AdvancedSearchRequestType advancedSearchRequestType = new AdvancedSearchRequestType();
advancedSearchRequestType.setClassIdentifier("Part");
advancedSearchRequestType.setCaseSensitive(false);
String criteria = "[Title Block.Number] contains 'P0' && " +
    "[Title Block.Description] is not null";
advancedSearchRequestType.setCriteria(criteria);
String attribute1 = "Title Block.Number";
String attribute2 = "Title Block.Description";
String attribute3 = "Title Block.Lifecycle Phase";
advancedSearchRequestType.setResultAttributes(new String[]{attribute1,
    attribute2, attribute3});
advancedSearchRequestType.setDisplayName("Search123");
AdvancedSearchResponseType advancedSearchResponseType =
    agileStub.advancedSearch(advancedSearchRequestType);
```

### Getting the Searchable Attributes

Operation - getSearchableAttributes

```
QueryGetSearchableAttributesRequestType queryGetSearchableAttributesRequestType = new
    QueryGetSearchableAttributesRequestType();
```

```
queryGetSearchableAttributesRequestType.setClassIdentifier("Part");  
QueryGetSearchableAttributesResponseType queryGetSearchableAttributesResponseType =  
agileStub.getSearchableAttributes(queryGetSearchableAttributesRequestType)  
;
```



# Working with Folders

**This chapter includes the following:**

---

▪ Loading a Folder .....	90
▪ Creating a Folder .....	91
▪ Working with Folder Elements .....	91
▪ Deleting a Folder .....	92
▪ Renaming a Folder .....	93

There are several types of Agile PLM folders:

- **Private** – Folders that are accessible only to the user that created them. Users can create or delete their own Private folders.
- **Public** – Folders that are accessible to all Agile PLM users. Only users with the GlobalSearches privilege can create, delete, and modify Public folders.
- **System** – Predefined folders that ship with the Agile PLM system. Most users cannot modify or delete System folders.
- **My Bookmarks (or Favorites)** – A predefined folder containing each user's bookmarks to Agile PLM objects. You cannot delete the My Bookmarks folder.
- **Home** – The predefined Agile PLM home folder. You cannot delete the Home folder.
- **Personal Searches** – The predefined parent folder for each user's personal searches. You cannot delete the Personal Searches folder.
- **Recently Visited** – A predefined folder containing links to recently visited objects. The web service does not populate this folder. It is only populated by client applications. If required, you specify this in your application.

---

**Note** The recently visited folder is only flushed to the database periodically. Therefore, secondary connections like process extensions with portals, or standalone web service based applications will not see the same information that the user's GUI displays.

---

- **Report** – A folder containing reports. Although you cannot use the Agile API to create, modify, or delete report folders, you can create, modify, or delete them in Agile PLM clients.

Each user's selection of folders may vary. However, every user has a *Home* folder. From each user's Home folder, you can construct various subfolders and browse public and private queries.

Folders are subject to the same transactional model as other Agile API objects. If you do not set a transaction boundary for a folder, it is automatically updated as soon as you add anything to, or remove anything from the folder.

You can work with Folder Web services to deal with the hierarchical structure of a folder by adding and removing children, getting children, and getting the parent folder.

## Loading a Folder

There are two ways to load a folder:

- Use the `getFolder` operation to specify the full path of a folder.
- Use the `getChildNode` operation to specify the relative path of a subfolder.

Folder and query names are not case-sensitive. Therefore, you can specify a folder path using upper or lower case. For example, to load the Personal Searches folder, you can specify `/Personal Searches` or `/PERSONAL SEARCHES`.

The following example shows how to load a folder by specifying the full path to the folder.

### Example: Loading a folder using the `getFolder` operation

```
GetFolderRequestType getFolderRequestType =
    new GetFolderRequestType();
AgileGetFolderRequestType[] agileGetFolderRequestTypeArray =
    new AgileGetFolderRequestType[1];
agileGetFolderRequestTypeArray[0] =
    new AgileGetFolderRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3. Please refer to the
documentation
                                // for other base folder constants
identifier.setName("Personal Searches"); //relative name of the folder
agileGetFolderRequestTypeArray[0].setFolderIdentifier(identifier);
agileGetFolderRequestTypeArray[0].setRecursive(false); //set to true if you want to load
all child folders recursively
getFolderRequestType.setRequests(agileGetFolderRequestTypeArray);
GetFolderResponseType getFolderResponseType =
    agileFolderStub.getFolder(getFolderRequestType);
```

The following example shows how to load a folder by specifying its path relative to another folder, in this case the user's Home Folder.

### Example: Loading a folder using `getChildNode` operation

```
GetChildNodeRequestType getChildNodeRequestType =
    new GetChildNodeRequestType();
AgileGetChildNodeRequestType[] agileGetChildNodeRequestTypeArray =
    new AgileGetChildNodeRequestType[1];
agileGetChildNodeRequestTypeArray[0] =
    new AgileGetChildNodeRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3.
identifier.setName("Personal Searches/Folder A"); //relative name of the folder A
agileGetChildNodeRequestTypeArray[0].setFolderIdentifier(identifier);
String[] children = new String[2];
children[0] = "Folder 1";
children[1] = "Query 1";
agileGetChildNodeRequestTypeArray[0].setChildren(children);
getChildNodeRequestType.setRequests(agileGetChildNodeRequestTypeArray);
GetChildNodeResponseType getChildNodeResponseType =
    agileFolderStub.getChildNode(getChildNodeRequestType);
```



## Creating a Folder

To create a folder, use the `createFolder` operation. When you create a folder, you must specify the folder's name and its parent folder. The following example shows how to create a folder named "MyTemporaryQueries" in the Personal Searches folder.

### Example: Creating a new folder

```
CreateFolderRequestType createFolderRequestType =
    new CreateFolderRequestType();
AgileCreateFolderRequestType[] agileCreateFolderRequestTypeArray =
    new AgileCreateFolderRequestType[1];
agileCreateFolderRequestTypeArray[0] =
    new AgileCreateFolderRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3.
identifier.setName("Personal Searches/Folder A"); //relative name of the folder A
agileCreateFolderRequestTypeArray[0].setFolderIdentifier(identifier);
createFolderRequestType.setRequests(agileCreateFolderRequestTypeArray);
```

## Working with Folder Elements

An Agile PLM folder can contain file folder objects (subfolders), query objects, and any kind of data object, such as item, manufacturer, etc. Use the `addChildNode` operation to add objects to a folder.

The operation is used to add childNodes to a folder identified by its folder identifier. The list of childNodes to be added to a folder will be passed as input to the request. The child nodes can be a query object, an AgileObject or a subfolder itself.

## Adding Folder Elements

The following example shows how to add a child node to a folder.

### Example: Adding child nodes to a folder

```
AddChildNodeRequestType addChildNodeRequestType =
    new AddChildNodeRequestType();
AgileAddChildNodeRequestType[] agileAddChildNodeRequestTypeArray =
    new AgileAddChildNodeRequestType[1];
agileAddChildNodeRequestTypeArray[0] =
    new AgileAddChildNodeRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3
identifier.setName("Personal Searches/Folder A"); //relative name of the folder A
agileAddChildNodeRequestTypeArray[0].setFolderIdentifier(identifier);
FolderIdentifierType[] childFolders = new FolderIdentifierType[2];
for(int i=0; i<childFolders.length; i++) {
    childFolders[i] = new FolderIdentifierType();
    childFolders[i].setBaseFolder(3);
    childFolders[i].setName("Personal Searches/Folder "+i);
}
agileAddChildNodeRequestTypeArray[0].setFolders(childFolders);
addChildNodeRequestType.setRequests(agileAddChildNodeRequestTypeArray);
```

## Removing Folder Elements

To remove a single folder element, use the `removeChildNode` operation.

### Example: Removing child nodes from a Folder

```
RemoveChildNodeRequestType removeChildNodeRequestType =
    new RemoveChildNodeRequestType();
AgileRemoveChildNodeRequestType[] agileRemoveChildNodeRequestTypeArray =
    new AgileRemoveChildNodeRequestType[1];
agileRemoveChildNodeRequestTypeArray[0] =
    new AgileRemoveChildNodeRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3.
identifier.setName("Personal Searches/Folder A"); //relative name of the folder A
agileRemoveChildNodeRequestTypeArray[0].setFolderIdentifier(identifier);
String[] children = new String[2];
children[0] = "Folder 1";
children[1] = "Query 1";
agileRemoveChildNodeRequestTypeArray[0].setChildren(children);
removeChildNodeRequestType.setRequests(agileRemoveChildNodeRequestTypeArray);
```

## Getting Folder Elements

All objects contained in a folder, including subfolders, can be loaded by name. To retrieve an object from a folder, use the `getChildNode` operation. Remember, the object type for folder elements can vary. Depending on the object, you could be getting a subfolder, a query, or a dataobject, such as an `Item`.

### Example: Getting a folder element

```
GetChildNodeRequestType getChildNodeRequestType =
    new GetChildNodeRequestType();
AgileGetChildNodeRequestType[] agileGetChildNodeRequestTypeArray =
    new AgileGetChildNodeRequestType[1];
agileGetChildNodeRequestTypeArray[0] =
    new AgileGetChildNodeRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3.
identifier.setName("Personal Searches/Folder A"); //relative name of the folder A
agileGetChildNodeRequestTypeArray[0].setFolderIdentifier(identifier);
String[] children = new String[2];
children[0] = "Folder 1";
children[1] = "Query 1";
agileGetChildNodeRequestTypeArray[0].setChildren(children);
getChildNodeRequestType.setRequests(agileGetChildNodeRequestTypeArray);
```

## Deleting a Folder

To delete a folder, use the `deleteFolder` operation. You can delete folders that are empty and that are not predefined Agile PLM system folders (such as the Global Searches and My Inbox folders).

Unlike other dataobjects, folders are not “soft-deleted” the first time you delete them. When you delete a folder, it is removed permanently from the system.

### Example: Deleting a folder

```
DeleteFolderRequestType deleteFolderRequestType =
    new DeleteFolderRequestType();
```

```
AgileDeleteFolderRequestType[] agileDeleteFolderRequestTypeArray =
    new AgileDeleteFolderRequestType[1];
agileDeleteFolderRequestTypeArray[0] =
    new AgileDeleteFolderRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3.
identifier.setName("Personal Searches/Folder A"); //relative name of the folder A
agileDeleteFolderRequestTypeArray[0].setFolderIdentifier(identifier);
deleteFolderRequestType.setRequests(agileDeleteFolderRequestTypeArray);
```

## Renaming a Folder

To rename a folder, use `renameFolder` operation.

### Example: Renaming a Folder

```
RenameFolderRequestType renameFolderRequestType =
    new RenameFolderRequestType();
AgileRenameFolderRequestType[] agileRenameFolderRequestTypeArray =
    new AgileRenameFolderRequestType[1];
agileRenameFolderRequestTypeArray[0] =
    new AgileRenameFolderRequestType();
FolderIdentifierType identifier = new FolderIdentifierType();
identifier.setBaseFolder(3); // Home folder constant is 3.
identifier.setName("Personal Searches/Folder A"); //relative name of the folder A
agileRenameFolderRequestTypeArray[0].setFolderIdentifier(identifier);
agileRenameFolderRequestTypeArray[0].setNewName("Folder B");
renameFolderRequestType.setRequests(agileRenameFolderRequestTypeArray);
```



# Creating and Managing Projects

**This chapter includes the following:**

---

▪ About Projects and Projects Objects .....	95
▪ Differences in the Behavior of Projects Objects .....	95
▪ Working with Project Baselines .....	96
▪ Substituting Project Resources.....	96
▪ Delegating Ownership of a Project to Another User.....	96
▪ Locking and Unlocking Projects.....	97

## About Projects and Projects Objects

You can use the project management features of Agile Product Portfolio Management (PPM) to define a project and its associated elements such as activity schedules, deliverables, and discussions. These capabilities enable you to determine the availability of the required resources, assigning resources to tasks, identifying bottlenecks, and responding to over- and under-allocated resource conditions. You can also create and reuse project templates.

The Projects object is used to schedule and execute projects. Each project, in addition to schedule information, contains attachments, discussions and actions items, resources and roles, and history and content of related activities. For management visibility, data is rolled up to higher levels by rules and parent-child relationships.

The Agile Project web services provide support for creating, loading, and working with Projects. These handle all Projects objects, including programs, phases, tasks, and gates.

## Differences in the Behavior of Projects Objects

The project web service helps implement several interface commonly used by other Agile PLM objects. However, it also provides the following distinct functionality that separates Projects objects from other objects.

- The Projects object is a container of other underlying Projects objects, such as Phases, Tasks, and Gates. The underlying Projects objects are related to the parent object, usually the Projects, through the Schedule table.
- Projects have baselines that allow you to track changes in the schedule. Therefore, the project web service provides operation that let you create, get, or remove a baseline.
- Projects can be archived. If you archive the root Projects, the entire Projects tree is soft-deleted from the system.
- Projects can be locked or unlocked.

## Working with Project Baselines

Projects baselines allow you to compare actual progress with your original plans. When you create a baseline, a snapshot of your Projects' schedule is preserved. The original estimates contained in the baseline are permanent reference points against which you can compare the updated task structure, schedule, and actual dates.

Baselines can be created only for the root Projects object. You can save multiple baselines, and retrieve them later for comparison. The following web services operations are used for creating, retrieving, and removing baselines:

- `createBaseline`
- `getBaselines`
- `removeBaseline`
- `getCurrentBaseline`

### Example: Creating and retrieving baselines

```
try {
```

Once a Projects' schedule is defined, you can reschedule it using the `reschedule` operation. This operation takes a couple of parameters, the `IProgram.RESCHEDULE` constant and the new value for that schedule option. Here is the list of `IProgram.RESCHEDULE` constants you can use:

- `STARTDATE` – This moves the scheduled start date to the specified date.
- `ENDDATE` – This moves the scheduled end date to the specified date.
- `BACKWARD_DAYS` – This moves the schedule backward by the specified number of days.
- `FORWARD_DAYS` – This moves the schedule forward by the specified number of days.

### Example: Rescheduling Projects

```
try {
```

## Substituting Project Resources

A resource's availability can frequently change due to overloading, reassignments, vacation, and illness. You can substitute an existing resource for another resource. The current resource's role is assigned to the substituted resource, but only for that Projects. To substitute Projects resources, use the `substituteResource` operation.

When you substitute resources, you can specify users as well as user groups. You can also specify whether the resource assignment applies to the Projects' children.

### Example: Substituting Projects resources

```
try {
```

## Delegating Ownership of a Project to Another User

The owner or manager of a Projects object can assign the ownership of the Projects to other users

by delegating it. The delegated user receives a request that he can accept or decline. If he accepts, the delegated user becomes owner of the task. A delegated owner is automatically given the Projects Manager role for the delegated Projects object.

To delegate ownership of a Project, use the `delegateOwnership` operation. When you delegate ownership of a Project, you automatically update the Delegated Owner field, which is read-only. The `delegateOwnership` operation lets you specify whether delegated ownership also applies to the Projects' children.

## Locking and Unlocking Projects

The owner of Projects can lock or unlock the Projects object. When a Project is locked, its schedule cannot be modified. To lock or unlock a Project, use the `setLock` operation.

---

**Note** Projects are automatically locked when you use the Gantt Chart or the Microsoft Project integration functionality in Agile Web Client.

---

## Checking Project's Lock Status





# Publishing Documents

### This chapter includes the following:

---

▪ Document Publishing Web Services Operations .....	99
▪ Loading XML Schema of Objects .....	100
▪ Loading XML Data of Objects .....	100

During the life of a product, Agile PLM acquires, processes, and maintains an assortment of data related to the product. This data is used in many ways and for different requirements to expedite, manage, and control product development activities.

Agile PLM Document Publishing Web Services enable creating documents based on data maintained by Agile PLM. This PLM data also lends itself to use by content developers and authors to prepare documents such as Product Data sheets, Parts List, and Service Manuals.

Agile PLM Release 9.3.1 supports the dynamic publication of these documents with embedded PLM data and triggers configured in PLM's Event Framework. The Dynamic Document Publishing solution can be used in the following instances:

- Create new structured documents (Product Data sheets, Parts List, Service Manual)
- Create documents in the native document publishing tool
- Browse and insert PLM metadata and file contents into documents
- Create formatted reports from PLM objects, search results, and push selected rows to reporting tools (compliance report, pricing model, quality report)
- Push an object ID, selected or all search results to a report for formatting
- Modify the content that is shared by other documents already stored in PLM
- Update documents that reference modified content

## Document Publishing Web Services Operations

Agile PLM Web Services exposes the following operations to enable creating the necessary XML schema and data files:

- **loadXMLSchema** – This Web Service API returns an XML package that fully describes the attributes of the object. This Web Service is used to create XML schema files that are used by BI Publisher to create the Templates.

For example, if you use this Web Service against a subclass like ECO, it will tell BI Publisher all of the possible attributes for ECOs. This is useful to be able to work with all potential attributes for an object when creating a Template.

- **loadXMLData** – This Web Service API returns the actual data that is stored for an object in an XML package. This web service is used to retrieve the object data that is combined with the Template to create the output file. You can also use the saved output from this Web Service to test a Template in BI Publisher.

## Loading XML Schema of Objects

```
LoadXMLSchemaRequestType loadXMLSchemaRequestType =
    new LoadXMLSchemaRequestType();
AgileLoadXMLSchemaRequestType[] agileLoadXMLSchemaRequestType =
    new AgileLoadXMLSchemaRequestType[1];
agileLoadXMLSchemaRequestType[0] = new AgileLoadXMLSchemaRequestType();
//set the API Names of the classes for which you want to load the schema
String[] classIds = new String[1];
classIds[0] = "Part";
agileLoadXMLSchemaRequestType[0].setClassIdentifiers(classIds);
loadXMLSchemaRequestType.setRequests(agileLoadXMLSchemaRequestType);
```

## Loading XML Data of Objects

```
DocPublishing_BindingStub agileAdminStub = getDocPublishingService();
LoadXMLDataRequestType loadXMLDataRequestType =
    new LoadXMLDataRequestType();
AgileLoadXMLDataRequestType[] agileLoadXMLDataRequestType =
    new AgileLoadXMLDataRequestType[1];
agileLoadXMLDataRequestType[0] = new AgileLoadXMLDataRequestType();
String[] filter = new String[1];
filter[0] = "DefaultPartFilter"; // APIName of the ACS filter
agileLoadXMLDataRequestType[0].setFilterName(filter);
ObjectReferentIdType[] objectRef = new ObjectReferentIdType[1];
objectRef[0] = new ObjectReferentIdType();
objectRef[0].setClassIdentifier("Part"); //APIName of the class
objectRef[0].setObjectIdentifier("P0001"); //Object Name of the class to be loaded
agileLoadXMLDataRequestType[0].setObjectReferent(objectRef);
loadXMLDataRequestType.setRequests(agileLoadXMLDataRequestType);
```

# Performing Administrative Tasks

This chapter includes the following:

---

- Managing Users..... 101

## Managing Users

Users are data objects that you can create, like items and changes. Consequently, you can work with users directly without traversing the administrative node hierarchy. If you have the proper Agile PLM privileges, you can create, modify, and delete users. For example, you could create a program that periodically synchronizes Agile PLM users with data available from a corporate directory.

## Handling User Passwords

### Changing Login Password

```
ChangeLoginPasswordRequestType changeLoginPasswordRequestType =
    new ChangeLoginPasswordRequestType();
AgileChangeLoginPasswordRequestType[] agileChangeLoginPasswordRequestTypeArray =
    new AgileChangeLoginPasswordRequestType[1];
agileChangeLoginPasswordRequestTypeArray[0] =
    new AgileChangeLoginPasswordRequestType();
agileChangeLoginPasswordRequestTypeArray[0].setUserIdentifier("admin");
agileChangeLoginPasswordRequestTypeArray[0].setNewPassword("newPassword");
agileChangeLoginPasswordRequestTypeArray[0].setOldPassword("oldPassword");
changeLoginPasswordRequestType.setRequests(agileChangeLoginPasswordRequestTypeArray);
```

### Changing Approval Password

```
ChangeApprovalPasswordRequestType changeApprovalPasswordRequestType =
    new ChangeApprovalPasswordRequestType();
AgileChangeApprovalPasswordRequestType[] agileChangeApprovalPasswordRequestTypeArray =
    new AgileChangeApprovalPasswordRequestType[1];
agileChangeApprovalPasswordRequestTypeArray[0] =
    new AgileChangeApprovalPasswordRequestType();
agileChangeApprovalPasswordRequestTypeArray[0].setUserIdentifier("admin");
agileChangeApprovalPasswordRequestTypeArray[0].setNewPassword("agile1");
agileChangeApprovalPasswordRequestTypeArray[0].setOldPassword("agile");
changeApprovalPasswordRequestType.setRequests(agileChangeApprovalPasswordRequestTypeArray);
```

## Managing Transfer Authorities

### Creating Transfer Authority

```
CreateTransferAuthorityRequestType createTransferAuthorityRequestType =
```

```
new CreateTransferAuthorityRequestType();
AgileCreateTransferAuthorityRequestType[] agileCreateTransferAuthorityRequestTypeArray =
    new AgileCreateTransferAuthorityRequestType[1];
agileCreateTransferAuthorityRequestTypeArray[0] =
    new AgileCreateTransferAuthorityRequestType();
TransferAuthorityRequestType transferAuthorityRequestType = new
TransferAuthorityRequestType();
transferAuthorityRequestType.setFromUser("badriy"); //loginID of from user
transferAuthorityRequestType.setToUser("yvonnec"); //loginID of to user
Calendar fromDate = Calendar.getInstance();
Calendar toDate = (Calendar) fromDate.clone();
toDate.add(Calendar.DAY_OF_YEAR, + 5);
transferAuthorityRequestType.setFromDate(fromDate);
transferAuthorityRequestType.setToDate(toDate);
transferAuthorityRequestType.setCriteria("AllChanges"); // For an exhaustive list of
various criterion that may be used, refer to the Java Client
//This field has two possible input values corresponding to the types
'AFFECT_CHANGES_ALL' - 0 or AFFECT_CHANGES_IN_PERIOD -1
transferAuthorityRequestType.setAffectChanges(0);
agileCreateTransferAuthorityRequestTypeArray[0].setTransferAuthorityRecord(transferAuthor
ityRequestType);
createTransferAuthorityRequestType.setRequests(agileCreateTransferAuthorityRequestTypeArr
ay);
```

## Getting Transfer Authority

```
GetTransferAuthorityRequestType getTransferAuthorityRequestType =
    new GetTransferAuthorityRequestType();
//set to true if you want to retrieve all records
getTransferAuthorityRequestType.setAll(false);

//to download individual records set the unique transfer record Ids, also set the boolean
flag as false. See below.
//          getTransferAuthorityRequestType.setAll(false);
//          AgileGetTransferAuthorityRequestType[]
agileGetTransferAuthorityRequestTypeArray =
//          new AgileGetTransferAuthorityRequestType[1];
//          agileGetTransferAuthorityRequestTypeArray[0] = new
AgileGetTransferAuthorityRequestType();
//
agileGetTransferAuthorityRequestTypeArray[0].setTransferRecordId("6099367"); //A unique
Id.
//
getTransferAuthorityRequestType.setTransferRecords(agileGetTransferAuthorityRequestTypeAr
ray);
```

## Removing Transfer Authority

```
RemoveTransferAuthorityRequestType removeTransferAuthorityRequestType =
    new RemoveTransferAuthorityRequestType();
AgileRemoveTransferAuthorityRequestType[] agileRemoveTransferAuthorityRequestTypeArray =
    new AgileRemoveTransferAuthorityRequestType[1];
agileRemoveTransferAuthorityRequestTypeArray[0] = new
AgileRemoveTransferAuthorityRequestType();
agileRemoveTransferAuthorityRequestTypeArray[0].setTransferRecordId("6102717"); //A
unique Id.
removeTransferAuthorityRequestType.setRequests(agileRemoveTransferAuthorityRequestTypeArr
ay);
```

# Working with Recipe & Material Workspace

**This chapter includes the following:**

▪ Performing Searches .....	103
▪ Editing User Data.....	105
▪ Processing Business Objects .....	107
▪ Using cfmXML Schema .....	109

This chapter describes how to work with Business Objects Processing, Searches and Editing services in the Agile Recipe and Material Workspace (RMW) module and provides sample code snippets.

## Performing Searches

RMW searches can be simple or advanced searches, with multiple criteria. RMW search operation lets you perform searches and sorting in one go.

## Using Operators

This feature allows the user to provide the operator type along with the criteria. Only operators that are supported by the RMW UI are supported in the enhanced CfmXML.

Operators that are supported by enhanced CfmXML are:

- OP\_DEFAULT
- OP\_MATCHES
- OP\_BETWEEN

The Operators are supported with the following modification to CfmXML

- Optional Values element inside Attribute element. This will contain n number of Values element
- qOptional operator attribute in Attribute element with the following possible values

**Example: Searching by User using the OP\_DEFAULT Operator**

```
<SearchCriteria id="1">
  <SearchName>AdminUser</SearchName>
  <CategoryName>User</CategoryName>
  <CategoryDBName>D_PPL_USER</CategoryDBName>
  <ViewName/>
  <Attribute name="User ID" operator="OP_DEFAULT">
    <Value dataType="Text">Administrator</Value>
    <UnitOfMeasure/>
  </Attribute>
</SearchCriteria>
```

```
</SearchCriteria>
```

**Example: Searching between Dates using the OP\_BETWEEN Operator**

```
<SearchCriteria id="1">
  <CategoryName>User</CategoryName>
  <CategoryDBName>D_PPL_USER</CategoryDBName>
  <ViewName/>
  <Attribute name="Created Date" operator="OP_BETWEEN">
    <Value dataType=""/>
    <Values>
      <Value GMTDateTime="1156863952000"
dataType="Date">29/08/2006</Value>
      <Value GMTDateTime="1156863952000"
dataType="Date">29/08/2006</Value>
    </Values>
    <UnitOfMeasure/>
    <Format>MM/dd/yyyy</Format>
  </Attribute>
</SearchCriteria>
```

## Searching with Sorting

This feature has an impact on the order of the results that are displayed. The sorting criteria specified in request xml will be considered and displayed accordingly. The user can specify the field name that has to sort on and the order that should be displayed in ascending or descending order.

Sorting using more than one field name is supported; in which case the order value will be considered to resolve the sorting order.

The sorting feature is supported with the following changes to CfmXML:

- Optional SortGroup element containing n number of Attribute elements.
- Optional order attribute in Attribute element which contains the order of sorting. If order is not specified for the attribute then actual order will be considered.
- Optional ascending attribute in Attribute element with yes/no possible values. Default value is yes.

**Example: Searching with sorting**

```
<SearchCriteria id="1">
  <CategoryName>User</CategoryName>
  <CategoryDBName>D_PPL_USER</CategoryDBName>
  <ViewName/>
  <Attribute name="User ID">
    <Value dataType=""/>
    <Values>
      <Value dataType="Text">Administrator</Value>
      <Value dataType="Text">System</Value>
      <Value dataType="Text">boprocqc</Value>
    </Values>
    <UnitOfMeasure/>
  </Attribute>
  <SortGroup>
    <Attribute ascending="yes" name="Is Active" order="1">
      <Value dataType="Text"/>
      <UnitOfMeasure/>
    </Attribute>
    <Attribute ascending="no" name="First Name" order="2">
      <Value dataType="Text"/>
      <UnitOfMeasure/>
    </Attribute>
  </SortGroup>
</SearchCriteria>
```

```

        </Attribute>
    </SortGroup>
</SearchCriteria>

```

## Searching with multiple criteria values

This features allows the user to search by providing multiple values for search criteria. This considers as **'In clause'** and the results will include all the records which meets the values in the list.

```

<SearchCriteria id="1">
    <CategoryName>User</CategoryName>
    <CategoryDBName>D_PPL_USER</CategoryDBName>
    <ViewName/>
    <Attribute name="Work Phone">
        <Value dataType=""/>
        <Values>
            <Value dataType="Phone">14001</Value>
            <Value dataType="Phone">91-5643728990</Value>
            <Value dataType="Phone">91-5643728990</Value>
        </Values>
        <UnitOfMeasure> </UnitOfMeasure>
        <Format>dd-MM-yyyy</Format>
    </Attribute>
</SortGroup>
    <Attribute ascending="no" name="Work Phone" order="1">
        <Value dataType="Text"/>
        <UnitOfMeasure/>
    </Attribute>
</SortGroup>
</SearchCriteria>

```

## Editing User Data

This is a generic edit Web service. Using this, you can perform edit operation on any category by providing the payload value in cfmXML format.

To invoke the edit (update) operation successfully ensure that the following element has a valid value:

```

<Attribute name="User ID">
    <Value dataType="Basic Text">ionidea</Value>
    <UnitOfMeasure></UnitOfMeasure>
</Attribute>

```

### Input Type

In input XML, it can be specified whether the editing operation is for Insert or Update or Delete. It is specified using **<Object operation="I">** or **<Object operation="U">** or **<Object operation="D">**.

### Example: Inserting a User data

```

<Payload>
    <Object operation="I">
        <CategoryName>WSEditTest</CategoryName>
        <CategoryDBName>WSEditTest</CategoryDBName>
        <ViewName/>
        <CustomInfo/>
        <ObjectKey>
            <Attribute name="UserId">

```

```
        <Value dataType="Integer">3</Value>
        <UnitOfMeasure/>
    </Attribute>
</ObjectKey>
<ObjectDetail>
    <Attribute name="Experience">
        <Value dataType="Integer">3</Value>
        <UnitOfMeasure/>
    </Attribute>
    <Attribute name="Rank">
        <Value dataType="Integer">3</Value>
        <UnitOfMeasure/>
    </Attribute>
</ObjectDetail>
</Object>
</Payload>
```

**Example: Updating a User data**

---

**Note** Before invoking the Update Input XML service, you must invoke the insert operation.

---

```
<Payload>
  <Object operation="U">
    <CategoryName>WSEditTest</CategoryName>
    <CategoryDBName>WSEditTest</CategoryDBName>
    <ObjectKey>
      <Attribute name="Experience">
        <Value dataType="Integer">4</Value>
      </Attribute>
    </ObjectKey>
    <ObjectDetail>
      <Attribute name="Rank">
        <Value dataType="Integer">1</Value>
      </Attribute>
    </ObjectDetail>
  </Object>
</Payload>
```

**Example: Inserting and Updating a User data**

```
<Payload>
  <Object operation="I">
    <CategoryName>Application Setup</CategoryName>
    <CategoryDBName>D_INT_APPLICATION_SETUP</CategoryDBName>
    <ViewName>DEFAULT</ViewName>
    <ObjectKey>
      <Attribute name="Application Name">
        <Value dataType="Basic Text">App Insert Test</Value>
        <UnitOfMeasure/>
      </Attribute>
    </ObjectKey>
    <ObjectDetail>
      <Attribute name="Application User Id">
        <Value dataType="Basic Text">App Insert Test User Id</Value>
        <UnitOfMeasure/>
      </Attribute>
      <Attribute name="Inbound XSL">
        <Value dataType="Basic Text">App Insert Test Inbound
XSL</Value>
        <UnitOfMeasure/>
      </Attribute>
      <Attribute name="Outbound XSL">
```



```

                                <Value dataType="Basic Text">App Insert Test Outbound
XSL</Value>
                                <UnitOfMeasure/>
                                </Attribute>
                                </ObjectDetail>
                                </Object>
                                <Object operation="U">
                                    <CategoryName>Application Setup</CategoryName>
                                    <CategoryDBName>D_INT_APPLICATION_SETUP</CategoryDBName>
                                    <ViewName>DEFAULT</ViewName>
                                    <ObjectKey>
                                        <Attribute name="Application Name">
                                            <Value dataType="Basic Text">Test App Name</Value>
                                            <UnitOfMeasure/>
                                        </Attribute>
                                    </ObjectKey>
                                    <ObjectDetail>
                                        <Attribute name="Application User Id">
                                            <Value dataType="Basic Text">Updated User Id</Value>
                                            <UnitOfMeasure/>
                                        </Attribute>
                                    </ObjectDetail>
                                </Object>
                                </Payload>

```

## Processing Business Objects

This is a generic web service to process any Business Objects (BO) using Web Service. This web service is generic enough to process any BO configured in the system. Operation such as Add/Modify/Delete can be performed on Business Object by just invoking this web service with BO details in the form of cfmXML.

BO web service processes multiple BOs. Hence, there is a standard mechanism to report errors and warning, if any during the processing of BOs.

- **Response code: 300** - This response code indicates failure of processBO operation with errors, warnings and validations as part of response message.
- **Response code: 301** - This response code indicates failure of processBO operation and errors, warnings and validations are contained in the response message.
- **Response code: 500** - This response code indicates success of processBO operation.

### Example: Inserting a new Application

```

<Payload>
    <Object>
        <CategoryName/>
        <CategoryDBName/>
        <ObjectKey/>
        <ObjectDetail/>
    </Object>
    <ObjectGroup isBO="Yes" name="Application">
        <BOActions>
            <BOAction>New</BOAction>
        </BOActions>
        <Object name="BORoot">
            <CategoryName>Application Setup</CategoryName>
            <CategoryDBName>D_INT_APPLICATION_SETUP</CategoryDBName>
            <ViewName/>

```

```
<CustomInfo/>
<ObjectKey>
  <Attribute name="Application Name">
    <Value
dataType="Text">JUNIT_BO_WS_Test_Application_041</Value>
    <UnitOfMeasure/>
  </Attribute>
</ObjectKey>
<ObjectDetail>
  <Attribute name="Inbound XSL">
    <Value dataType="Text">inbound</Value>
    <UnitOfMeasure/>
  </Attribute>
  <Attribute name="Outbound XSL">
    <Value dataType="Text">outbound</Value>
    <UnitOfMeasure/>
  </Attribute>
</ObjectDetail>
</Object>
</ObjectGroup>
<ObjectGroup isBO="Yes" name="Application">
  <BOActions>
    <BOAction>New</BOAction>
  </BOActions>
  <Object name="BORoot">
    <CategoryName>Application Setup</CategoryName>
    <CategoryDBName>D_INT_APPLICATION_SETUP</CategoryDBName>
    <ViewName/>
    <CustomInfo/>
    <ObjectKey>
      <Attribute name="Application Name">
        <Value
dataType="Text">JUNIT_BO_WS_Test_Application_002</Value>
        <UnitOfMeasure/>
      </Attribute>
    </ObjectKey>
    <ObjectDetail>
      <Attribute name="Inbound XSL">
        <Value dataType="Text">inbound</Value>
        <UnitOfMeasure/>
      </Attribute>
      <Attribute name="Outbound XSL">
        <Value dataType="Text">outbound</Value>
        <UnitOfMeasure/>
      </Attribute>
    </ObjectDetail>
  </Object>
</ObjectGroup>
</Payload>
```

**Example: Removing a User**

```
<Payload>
  <Object>
    <CategoryName/>
    <CategoryDBName/>
    <ObjectKey/>
    <ObjectDetail/>
  </Object>
  <ObjectGroup isBO="Yes" name="User">
    <BOActions>
      <BOAction>Remove</BOAction>
    </BOActions>
```

```

        <Object name="BORoot">
            <CategoryName>User</CategoryName>
            <CategoryDBName>D_PPL_USER</CategoryDBName>
            <ViewName>DEFAULT</ViewName>
            <ObjectKey>
                <ObjectID>785538</ObjectID>
                <Attribute name="User ID">
                    <Value dataType="Basic
Text">Administrator_Test01</Value>
                    <UnitOfMeasure/>
                </Attribute>
            </ObjectKey>
            <ObjectDetail/>
        </Object>
    </ObjectGroup>
</Payload>

```

## Using cfmXML Schema

cfmXML is a proprietary XML standard developed by Conformia. It is used for communication within Scale-up Management System and between Scale-up Management System and other applications. Refer to Appendix A for a reference to the cfmXML schema.

Details of tags used in cfmXML are explained in this document with the help of few sample messages.

### Example: Sample cfmXML message 1

```

<?xml version="2.0" encoding="UTF-8"?>
<cfmXML>
    <LoginInfo>
        <UserName>String</UserName>
        <Password>String</Password>
        <Database>String</Database>
        <AppUserName>String</AppUserName>
        <AppPassword>String</AppPassword>
    </LoginInfo>
    <SessionInfo logout="true">
        <UserName>String</UserName>
        <SessionID>String</SessionID>
    </SessionInfo>
    <PageInfo>
        <PageName>String</PageName>
        <ReturnURL>
            <URL>String</URL>
        </ReturnURL>
    </PageInfo>
    <Response>
        <ResponseCode>String</ResponseCode>
        <ResponseMessage>String</ResponseMessage>
        <SessionID>String</SessionID>
        <RedirectURL>String</RedirectURL>
    </Response>
    <SearchCriteria/>
    <SearchInfo/>
    <Payload/>
    <AdditionalInfo/>
</cfmXML>

```

## Sample cfmXML message 1

```
<?xml version="2.0" encoding="UTF-8"?>
<cfmXML>
  <LoginInfo>
    <UserName>String</UserName>
    <Password>String</Password>
    <Database>String</Database>
    <AppUserName>String</AppUserName>
    <AppPassword>String</AppPassword>
  </LoginInfo>
  <SessionInfo logout="true">
    <UserName>String</UserName>
    <SessionID>String</SessionID>
  </SessionInfo>
  <PageInfo>
    <PageName>String</PageName>
    <ReturnURL>
      <URL>String</URL>
    </ReturnURL>
  </PageInfo>
  <Response>
    <ResponseCode>String</ResponseCode>
    <ResponseMessage>String</ResponseMessage>
    <SessionID>String</SessionID>
    <RedirectURL>String</RedirectURL>
  </Response>
  <SearchCriteria/>
  <SearchInfo/>
  <Payload/>
  <AdditionalInfo/>
</cfmXML>
```

### LoginInfo

This contains user authentication information. User ID and Password are used for authenticating the user.

cfmXML tag	Outbound Message	Inbound Message
<UserName>	User ID the current user logged into Scale-up Management System.	User ID of the user-sending message. Value must be specified. Used to login the specified user in Scale-up Management System database.
<Password>	The Scale-up Management System database login password for the current user.	The value is specified for login purpose. If the value is not specified, it returns error.
<Database>	Name of the database to connect the external application. This will usually be blank for an outgoing message.	Name of the database to which Scale-up Management System need to connect to in order to serve the request.
<AppUserName>	Name of user in application setup table for a particular application	Name of user in application setup table for a particular application

<AppPassword>	Password of user in application setup table for a particular application	Password of user in application setup table for a particular application
---------------	--	--

## SessionInfo

This contains session ID of a successfully created session. Once a user is authenticated, session info can be used for subsequent calls to connect to an earlier created session.

cfmXML tag	Outbound Message	Inbound Message
<SessionInfo logout>	It contains the current Scale-up Management System session information. This will be useful if the invoked Web service needs to connect back to Scale-up Management System to get some more data.	It denotes the Scale-up Management System session ID of the previous Scale-up Management System session to be reused. If the specified session ID has expired, a new session is created for the user. Attribute logout contains information on whether to continue the session or end the session. If it's True, end the session or else continue the session.
<UserName>	User ID of the current logged in user.	It is the user login ID for the session to be reused.
<SessionID>	Session ID of the current user.	Contains session ID for the session to be reused.

## PageInfo

This contains the information about the page to integrate with for UI integration.

cfmXML tag	Outbound Message	Inbound Message
<PageName>	Not Used	Contains name of the page, which needs to be launched when control is transferred to Scale-up Management System.
<URL>	This is the return URL picked up from the Application Setup class for the application to which punch-out is being done. If there is no value specified in the Setup class, URL of the current Scale-up Management System page is sent as the return URL	If Return URL is present, cfmXML payload is sent to the Return URL and user is redirected to the ReturnURL.

## Response

This is the response code and message for the requested operation.

cfmXML tag	Outbound Message	Inbound Message
<ResponseCode>	Contains the Response code generated by the external application.	Contains the Response code generated by Scale-up Management System. 500 for success and 300 for error.
<ResponseMessage>	Contains the response message.	Contains the response message for the corresponding code.
<SessionID>	Contains session ID for the session in use.	Contains session ID for the Scale-up Management System session that was created or re-used for the request.
<RedirectURL>	Contains the external application URL to which Scale-up Management System user should be redirected to for UI integration.	Contains the Scale-up Management System URL to which user should be redirected to for UI integration.

## Sample cfmXML message 2

```
<?xml version="2.0" encoding="UTF-8"?>
<cfmXML>
  <LoginInfo/>
  <SessionInfo/>
  <PageInfo/>
  <ScreenContext/>
  <Response/>
  <SearchCriteria isPrimary="yes" id="0" execute="yes">
    <CategoryName>String</CategoryName>
    <CategoryDBName>String</CategoryDBName>
    <ViewName>String</ViewName>
    <Attribute name="String">
      <Value dataType="String" asEnteredValue="String">String</Value>
      <UnitOfMeasure>String</UnitOfMeasure>
    </Attribute>
    <Relationship type="inbound" refid="String" name="String" resolve="yes">
      <CategoryName>String</CategoryName>
      <CategoryDBName>String</CategoryDBName>
      <ObjectKey>
        <ObjectID>String</ObjectID>
        <Attribute name="String">
          <Value dataType="String" asEnteredValue="String">String</Value>
          <UnitOfMeasure>String</UnitOfMeasure>
        </Attribute>
        <Relationship type="inbound" refid="String" name="String" resolve="yes">
          <CategoryName>String</CategoryName>
          <CategoryDBName>String</CategoryDBName>
          <ObjectKey>
```

```

        <ObjectID>String</ObjectID>
        <Attribute name="String">
            <Value dataType="String" asEnteredValue="String">String</Value>
            <UnitOfMeasure>String</UnitOfMeasure>
        </Attribute>
    </ObjectKey>
</Relationship>
</ObjectKey>
</Relationship>
</SearchCriteria>
<SearchInfo>
    <SearchCriteria/>
    <SearchCriteria/>
</SearchInfo>
<Payload/>
<AdditionalInfo/>
</cfmXML>

```

## SearchCriteria

It contains the search criteria to be executed in Scale-up Management System and result set to be returned to the caller. The search criteria contain attribute names and values to be used as search parameters, category to be searched upon, and the view to be used for search and results.

cfmXML tag	Outbound Message	Inbound Message
<SearchCriteria execute>	Not Used	This contains the search criteria to perform the search. The execute attribute contains the information on whether to execute the search criteria and show the result. If value is "yes", the search is executed.
<CategoryName>	Not Used	Contains the name of the category for performing search.
<CategoryDBName>	Not Used	Contains the database name of the category for performing search.
<ViewName>	Not Used	Contains the name of the view for performing search.
<Attribute>	Not Used	Attribute (explained below) represents one attribute to search upon with search value. Search criteria may have multiple attributes each containing a search value.
<Relationship type refid name resolve>	Not Used	It contains reference to another object and associated search values to be searched upon in the database. Type indicates if the relationship is inbound or outbound. Name specifies the name for the relationship as

		defined in the system. Refid and resolve are used for internal use cases by Scale-up Management System.
--	--	---

## Attribute

It is used to specify an object attribute. Usually, it represents value of a cell in the database.

cfmXML tag	Outbound Message	Inbound Message
<Attribute name>	Contains the information about the attribute. Name attribute contains the name of the attribute.	Contains the information about the attribute. Name attribute contains the name of the attribute.
<Value dataType asEnteredValue>	Contains the value of the attribute. Attribute dataType contains the data type of attribute. For example. String, Number and so on.  Value can have following syntax <<Value>>- For single value <<Value>> to <<Value>> à upper and lower bounds to <<Value>> à upper bound <<Value >> to à lower bound	Contains the value of the attribute (as stored in the system). Attribute dataType contains the data type of attribute. For example. String, Number and so on. asEnteredValue contains the value for the attribute as was entered by the user, if applicable.  Value can have following syntax <<Value>>- For single value <<Value>> to <<Value>> à upper and lower bounds to <<Value>> à upper bound <<Value >> to à lower bound
<UnitOfMeasure>	Contains the value for unit of measure for the attribute value.	Contains the value for unit of measure for the attribute value.
<Format>	Contains format of Date  It can have following values dd/mm/yyyy, dd/mm/yyyy hh:mm:ss	Contains format of Date  It can have following values dd/mm/yyyy, dd/mm/yyyy hh:mm:ss

## Relationship

Contains information about other objects related to the object being defined in the message.

cfmXML tag	Outbound Message	Inbound Message
<CategoryName>	Contains the category name of the related category.	Contains the category name of the related category.



<CategoryDBName>	Contains the database name of the category name of the related category.	Contains the database name of the category name of the related category.
<ObjectKey>	If the relationship is a PK, it contains the primary keys data from the related category. If the relationship is not a PK, ObjectKey is not needed.	If the relationship is a PK, it contains the primary keys data from the related category. If the relationship is not a PK, ObjectKey is not needed. Details of Object Key are explained in the next section.

## ObjectKey

Contains information about other objects related to the object being defined in the message.

cfmXML tag	Outbound Message	Inbound Message
<ObjectID>	Contains the unique ID in the system for this object. This is internal ID in Scale-up Management System.	Contains the unique ID in the system for this object. This is internal ID in Scale-up Management System.
<Attribute>	This can be zero or many. All these attributes indicate the primary keys of the object.	This can be one or many. All these attributes indicate the primary keys of the object.
<Relationship>	These are references to other objects which are part of the primary keys of the object. This can be zero or many. Relationships can be nested.	These are references to other objects which are part of the primary keys of the object. This can be zero or many. Relationships can be nested.

## SearchInfo

It contains one more <SearchCriteria>. It is used to indicate multiple search criteria. RMW will execute all the search criteria and combine all the results into one response message. The primary use case for *SearchInfo* lies with the export feature of RMW.

## Sample cfmXML message 3

```
<?xml version="1.0" encoding="UTF-8"?>
<cfmXML>
  <LoginInfo/>
  <SessionInfo/>
  <PageInfo/>
  <ScreenContext/>
  <Response/>
  <SearchCriteria/>
  <SearchInfo/>
  <Payload>
```

```
<Object name="name">
  <CategoryDBName>String</CategoryDBName>
  <ViewName>String</ViewName>
  <ObjectKey/>
  <ObjectDetail>
    <Attribute/>
    <Relationship/>
  </ObjectDetail>
  <Document type="attachment" fileName="String" mimeType="String">String</Document>
  <OldValues>
    <ObjectKey/>
    <ObjectDetail/>
  </OldValues>
  <AdditionalInfo>
    <Attribute/>
    <Attribute/>
  </AdditionalInfo>
</Object>
<ObjectGroup name="name">
  <Object/>
  <Object/>
</ObjectGroup>
</Payload>
<AdditionalInfo>
<AdditionalInfo>
</cfmXML>
```

## Payload

Payload contains data records. An Object inside payload represents one data record. Each object contains a key section for primary keys of the objects and detail section to store non-primary keys of the object. Object also contains relationship information about the related objects.

cfmXML tag	Outbound Message	Inbound Message
<Object>	Represents one object in the payload. There could be one or more objects in a payload.	Represents one object in the payload. There could be one or more objects in a payload.
<ObjectGroup>	Object Group contains one or more objects. This is similar to payload with the difference that object groups cannot be nested.  Each ObjectGroup represents objects which are part of single transactions. This is currently used for export and import feature of Scale-up Management System.	Object Group contains one or more objects. This is similar to payload with the difference that object groups cannot be nested.  Each ObjectGroup represents objects which are part of single transactions. This is currently used for export and import feature of Scale-up Management System.

## Object

Contains information related to single data object. Attribute operation contains information on what type operation to be performed on this object.

Operation attribute may have the following values:

**I** - for Insert operation.

**U** - for Update operation.

**D** - for Delete operation.

cfmXML tag	Outbound Message	Inbound Message
<CategoryName>	Contains the category name of this object.	Contains the category name of this object.
<CategoryDBName>	Contains the category database name of this object.	Contains the category database name of this object.
<ViewName>	Contains the view name of this object.	Contains the view name of this object.
<ObjectKey>	This contains the information on object's primary keys	This contains the information on object's primary keys
<ObjectDetail>	This contains the information on object's non primary keys. It can contain zero or more attributes as well as relationships.	This contains the information on object's non primary keys. It can contain zero or more attributes as well as relationships.
<Document>	Contains text or binary document information related to this object. There could be zero or more of such documents. XML parser ignores the content of this tag.	Contains text or binary document information related to this object. There could be zero or more of such documents. XML parser ignores the content of this tag.
<OldValue>	This is used when there is a change in the values of the object. It stores all the values before the change is/was applied to the object. It consists of object keys and details. It is currently used internally by Scale-up Management System.	This is used when there is a change in the values of the object. It stores all the values before the change is/was applied to the object. It consists of object keys and details. It is currently used internally by Scale-up Management System.

## AdditionalInfo

This contains information, which cannot be represented as part of any object. These are a message level and object level tag. Any number of Attribute tags can be specified under AdditionalInfo. These may constitute the information that is needed for the message receiving application for any special processing.

This section can be used extensively for various general use cases.

For example, the Weigh scale web service, will pass the Equipment Weighing Activity ID along with its response as AdditionalInfo tag with the name starting with `SESSION_DATA_` as shown in the following sample XML:

```
</AdditionalInfo>
<Attribute name="ACTIVITY_ID">
  <Value dataType="STRING">ACT_5463</Value>
  <UnitOfMeasure></UnitOfMeasure>
  <Format></Format>
</Attribute>
</AdditionalInfo>
```



# Core Operations

This section describes the Core Web Services Operations. Download Agile Web Services Schema Docs from [Oracle eDelivery Web Site](http://edelivery.oracle.com) edelivery.oracle.com for the details of all the operations.



## Admin and Metadata Web Services

This chapter includes the following:

▪ getAllClasses .....	121
▪ getSubClasses .....	123
▪ getNode .....	125
▪ getLists .....	127
▪ getAttributes .....	128
▪ getTableMetadata .....	130
▪ getAutoNumbers .....	132
▪ getUsers .....	134
▪ getUserGroups .....	136
▪ convertCurrency .....	137
▪ changeLoginPassword .....	138
▪ changeApprovalPassword .....	139
▪ createTransferAuthority .....	140
▪ getTransferAuthority .....	141
▪ modifyTransferAuthority .....	142
▪ removeTransferAuthority .....	143
▪ getAgileClass .....	144

### getAllClasses

**Service** To retrieve all Agile classes from Agile PLM system.

**Usage** The class filtering details are specified in the request object. A list of Agile classes retrieved as per the filter is obtained in the response.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAllClasses
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <level>ALL</level>
        </request>
      </getAllClasses>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAllClassesResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
```

```
</messageName xsi:nil="true"/>
<statusCode>SUCCESS</statusCode>
<class>
  <nodeId>931</nodeId>
  <apiName>ChangesBaseClass</apiName>
  <typeCLASS</type>
  <displayName>Changes</displayName>
  <abstractClass>true</abstractClass>
  <subClass>
    <nodeId>1450</nodeId>
    <apiName>ManufacturerOrdersClass</apiName>
    <typeSUBCLASS</type>
    <displayName>Manufacturer Orders</displayName>
  </subClass>
  <subClass>
    <nodeId>6000</nodeId>
    <apiName>ChangeOrdersClass</apiName>
    <typeSUBCLASS</type>
    <displayName>Change Orders</displayName>
  </subClass>
</class>
... <!-- additional classes -->
</response>
</getAllClassesResponse>
</soapenv:Body>
</soapenv:Envelope>
```

**See also**      [getSubClasses](#)



## getSubClasses

**Service** To retrieve all Agile subclasses for a given base class from Agile PLM system.

**Usage** The request object contains relevant details for the same. A list of Agile subclasses are obtained in the response.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSubClasses
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <subClassesRequest>
            <classIdentifier>Changes</classIdentifier>
          </subClassesRequest>
          <subClassesRequest>
            <classIdentifier>Items</classIdentifier>
          </subClassesRequest>
        </request>
      </getSubClasses>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSubClassesResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <subClassesResponse>
            <classIdentifier>Changes</classIdentifier>
            <classes>
              <nodeId>1450</nodeId>
              <apiName>ManufacturerOrdersClass</apiName>
              <type>SUBCLASS</type>
              <displayName>Manufacturer Orders</displayName>
              <abstractClass>true</abstractClass>
              <superClass>
                <nodeId>931</nodeId>
                <apiName>ChangesBaseClass</apiName>
                <type>CLASS</type>
                <displayName>Changes</displayName>
              </superClass>
            </classes>
            <classes>
              <nodeId>6000</nodeId>
              <apiName>ChangeOrdersClass</apiName>
              <type>SUBCLASS</type>
              <displayName>Change Orders</displayName>
              <abstractClass>true</abstractClass>
              <superClass>
                <nodeId>931</nodeId>
                <apiName>ChangesBaseClass</apiName>
                <type>CLASS</type>
                <displayName>Changes</displayName>
              </superClass>
            </classes>
          </subClassesResponse>
        </response>
      </getSubClassesResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

**See also**

```
</soapenv:Envelope>
```

getAllClasses

## getNode

**Service** To retrieve Agile nodes for a given node identifier.

**Usage** The request object contains relevant details for the same. The queried node is obtained in the response.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getNode
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <nodeRequest>
            <nodeIdentifier>5009</nodeIdentifier>
            <recursive>true</recursive>
          </nodeRequest>
        </request>
      </getNode>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getNodeResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <nodeResponse>
            <node>
              <nodeId>5009</nodeId>
              <apiName>AutoNumbers</apiName>
              <typeNODE</type>
              <displayName>AutoNumbers</displayName>
              <childNodes>
                <nodeId>990</nodeId>
                <apiName>ECONumber</apiName>
                <typeNODE</type>
                <displayName>ECO Number</displayName>
                <Properties>
                  <propertyId>30</propertyId>
                  <apiName>Name</apiName>
                  <displayName>Name</displayName>
                  <readOnly>false</readOnly>
                  <Name xsi:type="xs:string"
                    xmlns:xs="http://www.w3.org/2001/XMLSchema">ECO
Number</Name>
                </Properties>
                <Properties>
                  <propertyId>38</propertyId>
                  <apiName>Description</apiName>
                  <displayName>Description</displayName>
                  <readOnly>false</readOnly>
                  <Description xsi:type="xs:string"
                    xmlns:xs="http://www.w3.org/2001/XMLSchema">ECO Number</Description>
                </Properties>
              </childNodes>
            </node>
          </nodeResponse>
        </response>
      </getNodeResponse>
    </soapenv:Body>

```

```
</soapenv:Envelope>
```

**See also** [getAttributes](#), [getTableMetadata](#), [getLists](#)

## getLists

**Service** To retrieve Agile Lists for a given List Identifier.

**Usage** The request object contains relevant details for the same. The retrieved lists are obtained in the response.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getLists
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <listsRequest>
            <listIdentifier>PartCategory</listIdentifier>
          </listsRequest>
        </request>
      </getLists>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getListsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <listsResponse>
            <list>
              <id>311</id>
              <apiName>PartCategory</apiName>
              <displayName>Part Category</displayName>
              <typeSIMPLELIST</type>
              <value xsi:nil="true"/>
            </entry>
            <entry>
              <id>2</id>
              <apiName>ELECTRICAL</apiName>
              <typeSIMPLELIST</type>
              <value>Electrical</value>
            </entry>
            <entry>
              <id>1</id>
              <apiName>MECHANICAL</apiName>
              <typeSIMPLELIST</type>
              <value>Mechanical</value>
            </entry>
          </list>
          <listIdentifier>311</listIdentifier>
        </listsResponse>
      </response>
    </getListsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

**See also** [getNode](#), [getTableMetadata](#), [getAttributes](#)

## getAttributes

**Service** To retrieve Agile attributes for a particular Class and Attribute Identifier.

**Usage** The request object contains relevant details for the same. The retrieved list attributes are obtained in the response.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAttributes
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <attributesRequests>
            <classIdentifier>Specification</classIdentifier>
            <attributeIdentifier>description</attributeIdentifier>
          </attributesRequests>
        </request>
      </getAttributes>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAttributesResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <attributesResponses>
            <attributes>
              <nodeId>2000001968</nodeId>
              <apiName>description</apiName>
              <type>ATTRIBUTE</type>
              <displayName>Description</displayName>
              <dataType>2</dataType>
              <searchable>true</searchable>
              <visible>true</visible>
              <required>false</required>
              <maxLength>100</maxLength>
              <properties>
                <propertyId>1</propertyId>
                <apiName>AttType</apiName>
                <displayName>AttType</displayName>
                <readOnly>false</readOnly>
                <AttType xsi:type="xs:string"
                  xmlns:xs="http://www.w3.org/2001/XMLSchema"></AttType>
              </properties>
              <properties>
                <propertyId>3</propertyId>
                <apiName>Max System Length</apiName>
                <displayName>Max System Length</displayName>
                <readOnly>true</readOnly>
                <MaxSystemLength xsi:type="xs:string"
                  xmlns:xs="http://www.w3.org/2001/XMLSchema">500</MaxSystemLength>
              </properties>
              <relationalOperators>EQ</relationalOperators>
              <relationalOperators>NEQ</relationalOperators>
              <relationalOperators>ISNULL</relationalOperators>
              <relationalOperators>ISNOTNULL</relationalOperators>
              <relationalOperators>LIKE</relationalOperators>
            </attributes>
          </attributesResponses>
        </response>
      </getAttributesResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

```
        <relationalOperators>STARTSWITH</relationalOperators>
        <relationalOperators>NOTSTARTSWITH</relationalOperators>
        <relationalOperators>CONTAINS</relationalOperators>
        <relationalOperators>NOTCONTAINS</relationalOperators>
      </attributes>
      <classIdentifier>Specification</classIdentifier>
    </attributesResponses>
  </response>
</getAttributesResponse>
</soapenv:Body>
</soapenv:Envelope>
```

**See also**      [getLists](#), [getNode](#), [getTableMetadata](#)

## getTableMetadata

**Service** To retrieve the metadata information of an Agile table in the PLM system

**Usage** The request object contains classIdentifier and tableIdentifier. The table metadata information retrieved is obtained through the response.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getTableMetadata
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <tableIdentifier>807</tableIdentifier>
          </requests>
          <requests>
            <classIdentifier>ECO</classIdentifier>
            <tableIdentifier>808</tableIdentifier>
          </requests>
        </request>
      </getTableMetadata>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getTableMetadataResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <attributes>
              <nodeId>3669</nodeId>
              <apiName>list19</apiName>
              <typeATTRIBUTE</type>
              <displayName>List19</displayName>
              <dataType>4</dataType>
              <possibleValues>
                <id xsi:nil="true"/>
                <apiName>list19</apiName>
                <displayName>List19</displayName>
                <typeSIMPLELIST</type>
                <value xsi:nil="true"/>
              </possibleValues>
              <searchable>true</searchable>
              <visible>false</visible>
              <required>false</required>
              <maxLength>2147483647</maxLength>
              <properties>
                <propertyId>1</propertyId>
                <apiName>AttType</apiName>
                <displayName>AttType</displayName>
                <readOnly>false</readOnly>
                <AttType xsi:type="xs:string"
                  xmlns:xs="http://www.w3.org/2001/XMLSchema"></AttType>
              </properties>
              <properties>
                <propertyId>5</propertyId>
                <apiName>DefaultValue</apiName>
                <displayName>DefaultValue</displayName>
              </properties>
            </attributes>
          </responses>
        </response>
      </getTableMetadataResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```



```
        <DefaultValue xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"/>
        </properties>
    </responses>
</response>
</getTableMetadataResponse>
</soapenv:Body>
</soapenv:Envelope>
```

**See also** [getAttributes](#)

## getAutoNumbers

**Service** To retrieve a suitable AutoNumber for an Agile object.

**Usage** The request object contains the Class and AutoNumber identifiers of the object. The AutoNumber for the object fetched by the Web Service is obtained through the response.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAutoNumbers
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <includeAllAutoNumberSource>true</includeAllAutoNumberSource>
            <size>3</size>
          </requests>
          <requests>
            <classIdentifier>ECO</classIdentifier>
            <includeAllAutoNumberSource>true</includeAllAutoNumberSource>
            <size>2</size>
          </requests>
        </request>
      </getAutoNumbers>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAutoNumbersResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <autoNumberResponses>
            <classIdentifier>Part</classIdentifier>
            <autoNumbers>
              <nodeId>12416</nodeId>
              <apiName>PartNumber</apiName>
              <type>AUTONUMBER</type>
              <displayName>Part Number</displayName>
              <autoNumber>P00580</autoNumber>
              <autoNumber>P00581</autoNumber>
              <autoNumber>P00582</autoNumber>
            </autoNumbers>
          </autoNumberResponses>
          <autoNumberResponses>
            <classIdentifier>ECO</classIdentifier>
            <autoNumbers>
              <nodeId>990</nodeId>
              <apiName>ECONumber</apiName>
              <type>AUTONUMBER</type>
              <displayName>ECO Number</displayName>
              <autoNumber>C00186</autoNumber>
              <autoNumber>C00187</autoNumber>
            </autoNumbers>
          </autoNumberResponses>
        </response>
      </getAutoNumbersResponse>
    </soapenv:Body>
```

```
</soapenv:Envelope>
```

**See also**`getAllClasses, getSubClasses`

## getUsers

**Service** To retrieve the information of Agile PLM Users.

**Usage** Obtains a list of users through the response object of the Web Service. The request object does not contain any element while the response consists of AgileUserType objects, which contain message elements carrying information pertaining to an Agile user.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getUsers
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns=""/>
      </getUsers>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getUsersResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <users>
            <objectIdentifier>
              <classId>11610</classId>
              <className>User</className>
              <classDisplayName>User</classDisplayName>
              <objectId>704</objectId>
              <objectName>admin</objectName>
            </objectIdentifier>
            <userID xsi:type="xs:string"
              xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11617"
              readOnly="False">admin</userID>
            <status xsi:type="common:AgileListEntryType"
              xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
              attributeId="12643" readOnly="True">
              <selection>
                <id>1</id>
                <apiName>ACTIVE</apiName>
                <value>Active</value>
              </selection>
            </status>
            <firstName xsi:type="xs:string"
              xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11614"
              readOnly="False">admin</firstName>
            <lastName xsi:type="xs:string"
              xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11616"
              readOnly="False">Administrator</lastName>
            <title xsi:type="xs:string"
              xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="12235"
              readOnly="False"></title>
            <address xsi:type="xs:string"
              xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11625"
              readOnly="False"></address>
            <geography xsi:type="common:AgileListEntryType"
              xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
              attributeId="8840" readOnly="False"/>
          </users>
        </response>
      </getUsersResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

```

        <city xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11626"
readOnly="False"></city>
        <postalZipCode xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11629"
readOnly="False"></postalZipCode>
        <businessPhone xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11619"
readOnly="False"></businessPhone>
        <homePhone xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11620"
readOnly="False"></homePhone>
        <mobilePhone xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11621"
readOnly="False"></mobilePhone>
        <fax xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11622"
readOnly="False"></fax>
        <pager xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11623"
readOnly="False"></pager>
        <email xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="11624"
readOnly="False">admin@admin.com</email>
        <secondaryEmail xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="12350"
readOnly="False"></secondaryEmail>
        <profile xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="8815"
readOnly="True"></profile>
        <roleS xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="11640" readOnly="False">
        </users>
    </getUsersResponse>
</soapenv:Body>
</soapenv:Envelope>

```

**See also**     `getUserGroups`

## getUserGroups

**Service** To retrieve the information of Users Groups in Agile PLM.

**Usage** Gets a list of user groups obtained through the response object of the Web Service. The request does not contain any element while the response consists of AgileUserGroupType objects, which carry message elements that contain information pertaining to an Agile user group.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getUserGroups
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns=""/>
      </getUserGroups>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getUserGroupsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </getUserGroupsResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getUsers](#)

## convertCurrency

<b>Service</b>	To convert a certain denomination and amount of money to a desired currency.
<b>Usage</b>	It converts a currency from one type to another given a certain date. The money is expressed as an object of type <code>AgileMoneyType</code> . The converted currency is obtained through the response object.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <convertCurrency
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <money>
              <amount>100.0</amount>
              <currency>INR</currency>
            </money>
            <toCurrency>GBP</toCurrency>
            <date>2009-05-05T13:37:35.555Z</date>
          </requests>
        </request>
      </convertCurrency>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <convertCurrencyResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <money>
              <amount>1.4843087362171332</amount>
              <currency>GBP</currency>
            </money>
            <date>2009-05-05T13:37:35.555Z</date>
          </responses>
        </response>
      </convertCurrencyResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

## changeLoginPassword

**Service** To change the login password for an Agile user.

**Usage** User identifier and new password information is supplied in the request. The details of the users whose passwords are to be changed, and the new passwords are specified in an array of requests. If the current user's password is to be reset then the old password must also be specified.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <changeLoginPassword
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <userIdentifier>yvonnec</userIdentifier>
            <oldPassword>agile1</oldPassword>
            <newPassword>agile</newPassword>
          </requests>
        </request>
      </changeLoginPassword>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <changeLoginPasswordResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </changeLoginPasswordResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```



## changeApprovalPassword

**Service** To change the approval password for an Agile user.

**Usage** User identifier and new password information is supplied in the request. The details of the users whose approval password is to be changed, along with their old and new passwords are specified in an array of requests. If the current user's approval password is to be reset then the old approval password must also be specified.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <changeApprovalPassword
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <userIdentifier>approval1</userIdentifier>
            <oldPassword>agile1</oldPassword>
            <newPassword>agile2</newPassword>
          </requests>
        </request>
      </changeApprovalPassword>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <changeApprovalPasswordResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </changeApprovalPasswordResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

## createTransferAuthority

**Service** To create a transfer authority record from an Agile user to another.

**Usage** The request contains information about the user from whom the authority is transferred and the user to whom the authority will be granted, the dates from and up to which the transfer is effective and the criteria identifying the authority to be transferred.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createTransferAuthority
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <transferAuthorityRecord>
              <fromUser>yvonnec</fromUser>
              <toUser>admin</toUser>
              <fromDate>2010-08-24T13:26:40.654Z</fromDate>
              <toDate>2010-09-23T13:26:40.654Z</toDate>
              <criteria>AllChanges</criteria>
              <affectChanges>1</affectChanges>
            </transferAuthorityRecord>
          </requests>
        </request>
      </createTransferAuthority>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createTransferAuthorityResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </createTransferAuthorityResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## getTransferAuthority

**Service** To retrieve a transfer authority record from the transfer authority record manager..

**Usage** A criteria identifying the authority to be retrieved is specified in the request. The request object does not contain any input element. All the transfer authority records found in the transfer authority manager are retrieved.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getTransferAuthority
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <all>true</all>
        </request>
      </getTransferAuthority>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getTransferAuthorityResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <transferAuthorityRecords>
            <transferRecordId>6099243</transferRecordId>
            <creator>admin</creator>
            <fromUser>yvonnec</fromUser>
            <toUser>admin</toUser>
            <fromDate>2010-06-01T11:45:49.000Z</fromDate>
            <toDate>2010-07-01T11:45:48.000Z</toDate>
            <criteria>AllChanges</criteria>
            <affectedChanges>1</affectedChanges>
            <lastChangeUser>admin</lastChangeUser>
          </transferAuthorityRecords>
          <transferAuthorityRecords>
            <transferRecordId>6099242</transferRecordId>
            <creator>admin</creator>
            <fromUser>yvonnec</fromUser>
            <toUser>admin</toUser>
            <fromDate>2010-06-01T11:38:13.000Z</fromDate>
            <toDate>2010-07-01T11:38:08.000Z</toDate>
            <criteria>AllChanges</criteria>
            <affectedChanges>1</affectedChanges>
            <lastChangeUser>admin</lastChangeUser>
          </transferAuthorityRecords>
        </response>
      </getTransferAuthorityResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

## modifyTransferAuthority

**Service** To retrieve the transfer authority record from the transfer authority record manager..

**Usage** A criteria identifying the authority to be retrieved is specified in the request. The request object does not contain any input element. All the transfer authority records found in the transfer authority manager are retrieved.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <ModifyTransferAuthority
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <transferAuthorityID>6103630</transferAuthorityID>
            <transferAuthorityRecord>
              <fromUser>yvonnec</fromUser>
              <toUser>admin</toUser>
              <fromDate>2012-05-11T18:30:00.000Z</fromDate>
              <toDate>2012-06-10T18:30:00.000Z</toDate>
              <criteria>AllChanges</criteria>
              <affectChanges>0</affectChanges>
            </transferAuthorityRecord>
          </requests>
        </request>
      </ModifyTransferAuthority>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <ModifyTransferAuthorityResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </ModifyTransferAuthorityResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## removeTransferAuthority

**Service** To remove a transfer authority record from the transfer authority record manager.

**Usage** A criteria identifying the authority to be retrieved is specified in the request. It contains an array of requests to support batch operations. The transfer authorities that are to be removed are specified in these requests.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeTransferAuthority
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <transferRecordId>6099247</transferRecordId>
          </requests>
        </request>
      </removeTransferAuthority>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeTransferAuthorityResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </removeTransferAuthorityResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## getAgileClass

**Service** To retrieve Agile classes from the PLM system.

**Usage** Details of the class whose subclasses are to be retrieved is specified in these requests. It contains an array of requests named `AgileGetClassRequestType` to support the batch operations. A `ClassType` with the details of the Agile class is obtained in the response.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAgileClass
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>PartsClass</classIdentifier>
          </requests>
        </request>
      </getAgileClass>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getAgileClassResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/AdminMetadata/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <agileClass>
              <nodeId>10141</nodeId>
              <apiName>Part</apiName>
              <type>SUBCLASS</type>
              <displayName>Part</displayName>
              <abstractClass>false</abstractClass>
              <superClass>
                <nodeId>10000</nodeId>
                <apiName>PartsClass</apiName>
                <type>CLASS</type>
                <displayName>Parts</displayName>
              </superClass>
            </agileClass>
          </responses>
        </response>
      </getAgileClassResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```







## Attachment Web Services

This chapter includes the following:

▪ getFileAttachment.....	147
▪ addFileAttachment.....	149
▪ checkOutAttachment .....	150
▪ checkInAttachment.....	151
▪ getFileFF.....	152
▪ addFileFF.....	153
▪ checkOutFF .....	154
▪ checkInFF .....	155
▪ cancelCheckOutFF.....	156

### getFileAttachment

**Service** To retrieve a specific file from the Attachments Tab of a particular Agile object.

**Usage** The request object contains the specifications that identify the attachment to be downloaded. An array of bytes is obtained in the response object, which also provides comprehensive information about the file retrieved.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getFileAttachment
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00734</objectNumber>
            <allFiles>false</allFiles>
            <downloadUrl>true</downloadUrl>
            <attachments>
              <rowId>6112830</rowId>
            </attachments>
          </requests>
        </request>
      </getFileAttachment>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getFileAttachmentResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </getFileAttachmentResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

```
<classIdentifier>Part</classIdentifier>
<objectNumber>P00734</objectNumber>
<attachment>
  <rowId>6112830</rowId>
  <fileId>6112635</fileId>
  <name>P00734 file123.txt</name>
  <description>Description for file 1</description>
  <fileType>txt</fileType>
  <fileSize>19</fileSize>
  <fileDownloadURL>http://DTP-VSREEDHA-
WF:8877/webfs/DownloadServlet?token=D6A8C1A41AA40B5AE29A2CFD33D0BEE143E7ABBBBC00245
67A16EBD62A195193C8FE6B3FBCC3131B58BC18A774412F759D648C6D0EC5D579A9E0B660217EA744A
24220B54E0583A1C569F6E9722B6&vault=&fileID=3DB9227A1EFF6DC2EB</fileDownloa
dURL>
</attachment>
</responses>
</response>
</getFileAttachmentResponse>
</soapenv:Body>
</soapenv:Envelope>
```

**See also**      loadTable, getRowId Method on page 285, getFileId Method on page 287

## addFileAttachment

**Service** To add a new file to the Attachment Tab of an Agile Object.

**Usage** This is facilitated by specifying relevant details of the new file through the request object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addFileAttachment
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00720</objectNumber>
            <singleFolder>false</singleFolder>
            <attachments>
              <name>Filename.txt</name>
              <description>Description for file </description>
              <content>RmlsZSBDb250ZW50Li4uZmlsZQ==</content>
            </attachments>
          </requests>
        </request>
      </addFileAttachment>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addFileAttachmentResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00720</objectNumber>
          </responses>
        </response>
      </addFileAttachmentResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** AddFileSOAPAttachment Method on page 283

## checkOutAttachment

**Service** To check out a specific file from the Attachment Tab of an Agile object and to retrieve its contents.

**Usage** The request object specifies the object and file to be retrieved while the desired content is received through the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkOutAttachment
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00726</objectNumber>
            <attachments>
              <rowId>6112534</rowId>
            </attachments>
          </requests>
        </request>
      </checkOutAttachment>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkOutAttachmentResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00726</objectNumber>
            <files>
              <rowId>6112534</rowId>
              <fileId>6112352</fileId>
              <fileType>txt</fileType>
              <fileSize>19</fileSize>
              <name>null File123.txt</name>
              <description>Description for file 1</description>
              <content>RmlsZSBDb250ZW50Li4uZmlsZQ==</content>
            </files>
          </responses>
        </response>
      </checkOutAttachmentResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** loadTable, checkInAttachment, getFileId Method on page 287, getRowId Method on page 285,

## checkInAttachment

**Service** To check-in an attachment that was previously checked out

**Usage** This attachment could possibly have been modified after checking out and has to be checked in back into the Agile system. Details of the modified file and its content, and the parent object, are specified in the request object. It is to be noted that the file name of the file being checked in must have the same extension as that of the file that was checked out.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkInAttachment
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <request>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00724</objectNumber>
            <rowId>6112462</rowId>
            <attachments>
              <fileName>Modified P00724 File123.txt</fileName>

<fileContent>TW9kaWZpZWQgZmlsZSBpbmZvcmlhdGlvb2IhZGRlZCBhZnRlciB0aGUy2h1Y2tpbg==</fileContent>
            </attachments>
          </request>
        </request>
      </checkInAttachment>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkInAttachmentResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </checkInAttachmentResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** checkOutAttachment, getFileId Method on page 287, getRowId Method on page 285

## getFileFF

**Service** To retrieve a list of files from a specific Agile file folder object.

**Usage** The request object contains the specifications that identify the file to be downloaded. An array of bytes is obtained in the response object, which also provides comprehensive information about the files retrieved, including content, file type, size, and row identifiers.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getFileFF
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <requests>
            <folderNumber>FOLDER00232</folderNumber>
            <folderVersion xsi:nil="true"/>
            <files>
              <rowId>6112773</rowId>
            </files>
          </requests>
        </request>
      </getFileFF>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getFileFFResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folderNumber>FOLDER00232</folderNumber>
            <folderVersion xsi:nil="true"/>
            <files>
              <rowId>6112773</rowId>
              <fileId>6112630</fileId>
              <name>FOLDER00232 File123.txt</name>
              <description>Description for file 1</description>
              <fileType>txt</fileType>
              <fileSize>19</fileSize>
              <content>RmlsZSBDb250ZW50Li4uZmlsZQ==</content>
            </files>
          </responses>
        </response>
      </getFileFFResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [loadTable](#), [getRowId Method](#) on page 285

## addFileFF

**Service** To add a new file to the Files Tab of an Agile File Folder object.

**Usage** The request object specifies details of the file folder that was previously checked out to facilitate the add file process. Before adding a new file to a folder, the folder object must be checked out prior to any file operation.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addFileFF
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <request>
            <folderNumber>FOLDER00220</folderNumber>
            <files>
              <fileName>File FOLDER00220.txt</fileName>
              <fileContent>RmlsZSBDb250ZW50Li4uZmlsZQ==</fileContent>
              <description>Description for file 1</description>
            </files>
          </request>
        </request>
      </addFileFF>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addFileFFResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </addFileFFResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [checkOutFF](#)

## checkOutFF

**Service** To check-out an Agile File Folder object.

**Usage** The request object specifies the file folder that has to be checked out. Subsequent operations such as adding a file to this file folder and then checking in the folder back are possible after this step. Before adding a new file to a folder, the folder object must be checked out prior to any file operation. The checkout Web Service is used to achieve this.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkOutFF
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <requests>
            <folderNumber>FOLDER00214</folderNumber>
          </requests>
        </request>
      </checkOutFF>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkOutFFResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folderNumber>FOLDER00214</folderNumber>
          </responses>
        </response>
      </checkOutFFResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getRowId Method on page 285](#), [loadTable](#)



## checkInFF

**Service** To check-in an Agile File Folder Object.

**Usage** The request object specifies the file folder that has already been checked out and needs to be checked in by the Web Service operation.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkInFF
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <requests>
            <folderNumber>FOLDER00220</folderNumber>
          </requests>
        </request>
      </checkInFF>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkInFFResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folderNumber>FOLDER00220</folderNumber>
          </responses>
        </response>
      </checkInFFResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** getRowId Method on page 285, checkOutFF

## cancelCheckOutFF

**Service** To cancel the 'checked-out' status of an Agile File Folder object that was earlier checked out using the checkout operation.

**Usage** The request object specifies the file folder for which the 'checked-out' status has to be annulled.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <cancelCheckOutFF
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <request xmlns="">
          <requests>
            <folderNumber>FOLDER00217</folderNumber>
          </requests>
        </request>
      </cancelCheckOutFF>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <cancelCheckOutFFResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Attachment/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folderNumber>FOLDER00217</folderNumber>
          </responses>
        </response>
      </cancelCheckOutFFResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [checkOutFF](#)





## Business Web Services

This chapter includes the following:

▪ createObject .....	159
▪ getObject .....	162
▪ updateObject .....	164
▪ deleteObject .....	165
▪ undeleteObject .....	166
▪ isDeletedObject .....	167
▪ sendObject .....	168
▪ saveAsObject .....	169
▪ checkPrivilege .....	170
▪ getSubscriptions .....	171
▪ modifySubscriptions .....	172

### createObject

**Service** To create a specific Object in Agile PLM system.

**Usage** The object specifications are detailed in the request object where the class type, unique object number and other primary data may be configured, apart from more specific options for the created object.

#### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <data rowId="0">
              <number>P00585</number>
              <description>Object Desc</description>
            </data>
          </requests>
        </request>
      </createObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
        </response>
      </createObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

```

<!---End--></responses>
<agileObject>
  <objectIdentifier>
    <classId>10141</classId>
    <className>Part</className>
    <classDisplayName>Part</classDisplayName>
    <objectId>6110466</objectId>
    <objectName>P00585</objectName>
  </objectIdentifier>
  <number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1001"
readOnly="False">P00585</number>
  <itemType xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1081" readOnly="False">
    <selection>
      <id>10141</id>
      <apiName>PART</apiName>
      <value>Part</value>
    </selection>
  </itemType>
  <lifecyclePhase xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1084" readOnly="True">
    <selection>
      <id>976</id>
      <apiName>PRELIMINARY</apiName>
      <value>Preliminary</value>
    </selection>
  </lifecyclePhase>
  <description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1002"
readOnly="False">Object Desc</description>
  <itemCategory xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1082" readOnly="False">
    <size xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1068" readOnly="False">
      <productLineS xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1004" readOnly="False">
        <rev xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1014" readOnly="False">
          <selection>
            <id>0</id>
            <apiName>Rev</apiName>
            <value>Introductory</value>
          </selection>
        </rev>
        <revIncorpDate xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1017" readOnly="True"/>
        <revReleaseDate xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1016" readOnly="True"/>
        <effectivityDate xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="12089" readOnly="True"/>
        <shippableItem xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000002781" readOnly="False">
          <selection>
            <id>0</id>
            <apiName>NO</apiName>
            <value>No</value>
          </selection>
        </shippableItem>
        <excludeFromRollup xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000002859" readOnly="False">
          <selection>
            <id>0</id>
            <apiName>NO</apiName>
            <value>No</value>
          </selection>
        </excludeFromRollup>
        <complianceCalculatedDate xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2000004143"
readOnly="True"/>
      </productLineS>
    </size>
  </itemCategory>
  <Object Desc</description>
  </itemType>
</agileObject>

```

```

        <partFamily xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000004412" readOnly="False"/>
        <mass xsi:type="common:AgileUnitOfMeasureType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000004612" readOnly="False"/>
        <overallCompliance xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000004891" readOnly="True"/>
        <itemGroupS xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000008520" readOnly="True"/>
        <thumbnail xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000008549" readOnly="True"/>
    </agileObject>
</responses>
</response>
</createObjectResponse>
</soapenv:Body>
</soapenv:Envelope>

```

**See also**      `getAutoNumbers`, `getAttributes`, `getAllClasses`, `getSubClasses`

## getObject

**Service** To retrieve a specific Agile object from the Agile PLM system.

**Usage** The specifications of the object to be retrieved are detailed in the request object where the class type, unique object number and relevant data may be specified. Comprehensive information about the object is retrieved in the response object after successful execution of the Web Service call.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getObject xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>ManufacturerPart</classIdentifier>
            <objectNumber>MANUF PART1241535324230</objectNumber>
            <tableRequests>
              <tableIdentifier>807</tableIdentifier>
              <loadCellMetaData>>false</loadCellMetaData>
            </tableRequests>
            <options>
              <propertyName>manufacturer name</propertyName>
              <propertyValue>MANUF1241535323652</propertyValue>
            </options>
          </requests>
        </request>
      </getObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <agileObject>
              <objectIdentifier>
                <classId>1488</classId>
                <className>ManufacturerPart</className>
                <classDisplayName>Manufacturer Part</classDisplayName>
                <objectId>6110515</objectId>
                <objectName>MANUF PART1241535324230</objectName>
              </objectIdentifier>
              <table>
                <tableIdentifier>
                  <classId>1488</classId>
                  <className>ManufacturerPart</className>
                  <objectId>6110515</objectId>
                  <objectName>MANUF PART1241535324230</objectName>
                  <tableId>807</tableId>
                  <tableName>Attachments</tableName>
                  <tableDisplayName>Attachments</tableDisplayName>
                </tableIdentifier>
              </table>
              <manufacturerPartNumber xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                attributeId="1648">MANUF_PART1241535324230</manufacturerPartNumber>
            </agileObject>
          </responses>
        </response>
      </getObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```



```

        <manufacturerName xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
attributeId="1649">
        <description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
attributeId="3566">Description</description>
        <lifecyclePhase xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1649">
        <selection>
        <id>1517</id>
        <apiName>ACTIVE</apiName>
        <value>Active</value>
        </selection>
        </lifecyclePhase>
        <mfrPartType xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2543">
        <selection>
        <id>1488</id>
        <apiName>MANUFACTURER PART</apiName>
        <value>Manufacturer Part</value>
        </selection>
        </mfrPartType>
        <partFamily xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000004417"/>
        <mass xsi:type="common:AgileUnitOfMeasureType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000004613"/>
        <complianceCalculatedDate xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2000004735"/>
        <overallCompliance xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000005405"/>
        <thumbnail xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000008559"/>
        <itemGroupS xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="2000008566"/>
        </agileObject>
    </responses>
</response>
</getObjectResponse>
</soapenv:Body>
</soapenv:Envelope>

```

**See also**      `quickSearch`, `advancedSearch`, `createObject`

## updateObject

**Service** To update a specific object in the Agile PLM system.

**Usage** The revised object specifications are detailed in the request object where data specific to an Agile object may be expressed.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <updateObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>ManufacturerPart</classIdentifier>
            <objectNumber>MANUF PART1241535380057</objectNumber>
            <data rowId="0">
              <Message Desc attributeId="3566">Updated value of Manuf part
Description</Message Desc>
            </data>
            <options>
              <propertyName>manufacturer name</propertyName>
              <propertyValue>MANUF1241535379620</propertyValue>
            </options>
          </requests>
        </request>
      </updateObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <updateObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </updateObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getAttributes](#)

## deleteObject

**Service** To delete a specific object in the Agile PLM system.

**Usage** The specifications of the object to be deleted are given in the request object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00589</objectNumber>
          </requests>
        </request>
      </deleteObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <isDeleted>true</isDeleted>
          </responses>
        </response>
      </deleteObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [quickSearch](#), [advancedSearch](#)

## undeleteObject

**Service** To revoke the 'deleted' status of a specific object that was previously deleted from the Agile PLM system.

**Usage** The specifications of the object to be undeleted are given in the request object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <undeleteObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00600</objectNumber>
          </requests>
        </request>
      </undeleteObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <undeleteObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses/>
        </response>
      </undeleteObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [advancedSearch](#), [quickSearch](#), [deleteObject](#)

## isDeletedObject

<b>Service</b>	To check whether a specific Agile object in the Agile PLM system has been deleted or not.
<b>Usage</b>	The object specifications are detailed in the request object where the class type, unique object number may be specified. From the response object, it is possible to ascertain whether the Agile object queried for still exists in the Agile PLM or if it was previously deleted.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <isDeletedObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00593</objectNumber>
          </requests>
        </request>
      </isDeletedObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <isDeletedObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <isDeleted>false</isDeleted>
          </responses>
        </response>
      </isDeletedObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

**See also**      [advancedSearch](#), [quickSearch](#), [deleteObject](#)

## sendObject

**Service** To send a specific Agile object to the Agile PLM system.

**Usage** The object specifications of the object to be sent are detailed in the request object.

☞ To send the objects through Web Services or web user interface, you must first enable the notification feature. In Java client, under Admin > Server settings > Database, set the Notification Enabled to Yes.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <sendObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00599</objectNumber>
            <sendTo>
              <classIdentifier>User</classIdentifier>
              <objectIdentifier>User1241535366448</objectIdentifier>
            </sendTo>
            <comments>Test comments</comments>
          </requests>
        </request>
      </sendObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <sendObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses/>
        </response>
      </sendObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getUsers](#)

## saveAsObject

<b>Service</b>	To save a specific Agile object as a new object in the Agile PLM system.
<b>Usage</b>	The object specifications are detailed in the request object where the class type, unique object number and other primary data may be specified. The response object contains information identifying the object that was saved.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <saveAsObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <saveAsObjectRequest>
            <parentClassIdentifier>Part</parentClassIdentifier>
            <parentObjectNumber>P00594</parentObjectNumber>
            <newClassIdentifier>Part</newClassIdentifier>
            <data rowId="0">
              <Message Num attributeId="1001">P00596</Message Num>
            </data>
          </saveAsObjectRequest>
          <saveAsObjectRequest>
            <parentClassIdentifier>Part</parentClassIdentifier>
            <parentObjectNumber>P00595</parentObjectNumber>
            <newClassIdentifier>Part</newClassIdentifier>
            <autoNumberSource>Part Number</autoNumberSource>
          </saveAsObjectRequest>
        </request>
      </saveAsObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <saveAsObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <saveAsObjectResponse>
            <classIdentifier>Part</classIdentifier>
            <objectId>6110572</objectId>
            <objectNumber>P00596</objectNumber>
          </saveAsObjectResponse>
          <saveAsObjectResponse>
            <classIdentifier>Part</classIdentifier>
            <objectId>6110579</objectId>
            <objectNumber>P00598</objectNumber>
          </saveAsObjectResponse>
        </response>
      </saveAsObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

**See also**      [getAutoNumbers](#), [getAttributes](#)

## checkPrivilege

**Service** To check whether a specific Agile user holds the privileges to perform a specific action in the Agile PLM system.

**Usage** The user and privilege specifications are detailed in the request object. The request object confirms whether or not the specified agile user has the privilege to perform the Web Service operation.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkPrivilege
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <userIdentification>
              <userIdentifier>admin</userIdentifier>
            </userIdentification>
            <privilege>1</privilege>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P00585</objectNumber>
          </requests>
        </request>
      </checkPrivilege>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <checkPrivilegeResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <userIdentification>
              <userIdentifier>admin</userIdentifier>
            </userIdentification>
            <privilege>
              <privilege>1</privilege>
              <checkPrivilege>true</checkPrivilege>
              <classIdentifier>Part</classIdentifier>
              <objectNumber>P00585</objectNumber>
            </privilege>
          </responses>
        </response>
      </checkPrivilegeResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getUsers](#), [copyTable](#)



## getSubscriptions

**Service** To retrieve the subscriptions for a specific data object in the Agile PLM system.

**Usage** The data object specifications are detailed in the request object. The response object contains the details of the subscriptions.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSubscriptions
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>TEST PART5554188</objectNumber>
          </requests>
          <requests>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>TEST ECO3514179</objectNumber>
          </requests>
        </request>
      </getSubscriptions>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSubscriptionsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <objectReferent>
              <classIdentifier>Part</classIdentifier>
              <objectIdentifier>TEST PART7510476</objectIdentifier>
              <version>0</version>
            </objectReferent>
            <subscriptions>
              <id>1</id>
              <name>Lifecycle Phase Change</name>
              <enable>false</enable>
            </subscriptions>
            <subscriptions>
              <id>2</id>
              <name>Field Change</name>
              <enable>false</enable>
            </subscriptions>
            <subscriptions>
              <id>13</id>
              <name>Rev Change</name>
              <enable>false</enable>
            </subscriptions>
            <subscriptions>
              <id>3</id>
              <name>Add File</name>
              <enable>false</enable>
            </subscriptions>
            <subscriptions>
              <id>4</id>
              <name>Delete File</name>
            </subscriptions>
          </responses>
        </response>
      </getSubscriptionsResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

```
        </subscriptions>
        <subscriptions>
          <id>5</id>
          <name>Checkin File</name>
          <enable>false</enable>
        </subscriptions>
        <subscriptions>
          <id>6</id>
          <name>Checkout File</name>
          <enable>false</enable>
        </subscriptions>
        <subscriptions>
          <id>7</id>
          <name>Cancel Checkout File</name>
          <enable>false</enable>
        </subscriptions>
      </responses>
    </response>
  </getSubscriptionsResponse>
</soapenv:Body>
</soapenv:Envelope>
```

## modifySubscriptions

**Service** To retrieve the subscriptions for a specific data object in the Agile PLM system.

**Usage** The data object and subscriptions specifications are detailed in the request object. The response object will contain the details of the subscriptions..

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <modifySubscriptions
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>TEST PART6194052</objectNumber>
            <subscriptions>
              <id>2</id>
              <name>Field Change</name>
              <enable>>false</enable>
            </subscriptions>
            <applyToChildren>>false</applyToChildren>
          </requests>
        </request>
      </modifySubscriptions>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <modifySubscriptionsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Business/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <subscriptions>
              <id>1</id>
              <name>Lifecycle Phase Change</name>
              <enable>>false</enable>
            </subscriptions>
            <subscriptions>
              <id>2</id>
              <name>Field Change</name>
              <enable>>true</enable>
            </subscriptions>
            <subscriptions>
              <id>13</id>
              <name>Rev Change</name>
              <enable>>false</enable>
            </subscriptions>
            <subscriptions>
              <id>3</id>
              <name>Add File</name>
              <enable>>false</enable>
            </subscriptions>
            <subscriptions>
              <id>4</id>
              <name>Delete File</name>
              <enable>>false</enable>
            </subscriptions>
            <subscriptions>
              <id>5</id>
            </subscriptions>
          </responses>
        </response>
      </modifySubscriptionsResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

```
        <enable>false</enable>
      </subscriptions>
    </subscriptions>
    <id>6</id>
    <name>Checkout File</name>
    <enable>false</enable>
  </subscriptions>
  <subscriptions>
    <id>7</id>
    <name>Cancel Checkout File</name>
    <enable>false</enable>
  </subscriptions>
</responses>
</response>
</modifySubscriptionsResponse>
</soapenv:Body>
</soapenv:Envelope>
```





## Collaboration Web Services

This chapter includes the following:

▪ getWorkflows .....	177
▪ getStatus .....	179
▪ auditRObjct .....	181
▪ getApprovers .....	182
▪ changeStatus .....	183
▪ approveRObjct .....	184
▪ rejectRObjct .....	185
▪ setWorkFlow .....	186
▪ addApprovers .....	187
▪ removeApprovers .....	188
▪ commentRObjct .....	189
▪ getReviewers .....	190

### getWorkflows

**Service** To retrieve the valid workflows of an Agile routable object.

**Usage** When you create a new change, package, product service request, or quality change order, you must select a workflow. Otherwise, the object remains in an unassigned state and cannot progress through a workflow process.

Agile system can have multiple workflows defined for each type of routable object. To retrieve the valid workflows for an object, use getWorkflows service.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getWorkFlows
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <workflowRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00034</objectNumber>
          </workflowRequest>
        </request>
      </getWorkFlows>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getWorkFlowsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
```

```
</messageName xsi:nil="true"/>
<statusCode>SUCCESS</statusCode>
<workflowResponse>
  <identifier>
    <classId>6141</classId>
    <className>ECO</className>
    <classDisplayName>ECO</classDisplayName>
    <objectId>6128130</objectId>
    <objectName>C00034</objectName>
  </identifier>
  <workflow>
    <workflowId>3752</workflowId>
    <workflowName>DefaultChangeOrders</workflowName>
    <workflowDisplayName>Default Change
Orders</workflowDisplayName>
  </workflow>
</workflowResponse>
</response>
</getWorkFlowsResponse>
</soapenv:Body>
</soapenv:Envelope>
```

**See also**      [setWorkFlow](#)



## getStatus

**Service** To get the current and the next workflow status of an routable object.

**Usage** To determine the status of a change whether it's pending or released. The getStatus service returns a Status object.

Workflow functionalities that are made available to users for a particular routable object, depends on the status of the routable object and the user's privileges. The workflow actions available for a pending change are different from those for a released change.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getStatus
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <statusRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00034</objectNumber>
          </statusRequest>
        </request>
      </getStatus>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getStatusResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <statusResponse>
            <identifier>
              <classId>6141</classId>
              <className>ECO</className>
              <classDisplayName>ECO</classDisplayName>
              <objectId>6128130</objectId>
              <objectName>C00034</objectName>
            </identifier>
            <currentStatus>
              <statusId>3753</statusId>
              <statusName>Pending</statusName>
              <statusDisplayName>Pending</statusDisplayName>
            </currentStatus>
            <nextDefaultStatus>
              <statusId>3766</statusId>
              <statusName>Submitted</statusName>
              <statusDisplayName>Submitted</statusDisplayName>
            </nextDefaultStatus>
            <nextStatus>
              <statusId>3766</statusId>
              <statusName>Submitted</statusName>
              <statusDisplayName>Submitted</statusDisplayName>
            </nextStatus>
          </statusResponse>
        </response>
      </getStatusResponse>
    </soapenv:Body>
```

```
</soapenv:Envelope>
```

## auditRObject

**Service** To audit a routable object.

**Usage** At any point in the lifecycle of a Change, you can audit the Change to determine if any required entry cells are incomplete or the change violates any Agile Smart Rules.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <auditRObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <request>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00035</objectNumber>
            <auditRelease>true</auditRelease>
          </request>
        </request>
      </auditRObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <auditRObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <responses xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <response>
            <identifier>
              <classId>6141</classId>
              <className>ECO</className>
              <classDisplayName>ECO</classDisplayName>
              <objectId>6128165</objectId>
              <objectName>C00035</objectName>
            </identifier>
            <error>
              <exceptionId>60086</exceptionId>
              <message>The following required fields are missing : C00035:
Affected Items.New Rev: P00336</message>
            </error>
          </response>
        </responses>
      </auditRObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [changeStatus](#)

## getApprovers

**Service** To get a list of approvers or observers for Agile's routable objects.

**Usage** When a routable object is released in a workflow, it is either sent to a user for approval or for notification. A list of users is required to be selected and added for the workflow to begin. This list is obtained from Agile system by sending a getApprovers request.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getApprovers
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <approversRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00038</objectNumber>
            <statusIdentifier>CCB</statusIdentifier>
          </approversRequest>
        </request>
      </getApprovers>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getApproversResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <approversResponse>
            <identifier>
              <classId>6141</classId>
              <className>ECO</className>
              <classDisplayName>ECO</classDisplayName>
              <objectId>6128265</objectId>
              <objectName>C00038</objectName>
            </identifier>
          </approversResponse>
        </response>
      </getApproversResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getStatus](#)

## changeStatus

**Service** A general purpose service for changing the status of an Agile object.

**Usage** To submit, release, or cancel a change.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <changeStatus
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <changeStatusRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00033</objectNumber>
            <newStatusIdentifier>Submitted</newStatusIdentifier>
            <comment>Comments</comment>
            <password>agile</password>
            <auditRelease>false</auditRelease>
            <urgent>false</urgent>
            <notifyOriginator>true</notifyOriginator>
            <notifyChangeAnalyst>true</notifyChangeAnalyst>
            <notifyCCB>true</notifyCCB>
          </changeStatusRequest>
        </request>
      </changeStatus>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
HTTP/1.1 200 OK
Date: Fri, 10 Apr 2009 11:55:27 GMT
Server: Oracle-Application-Server-10g/10.1.3.4.0 Oracle-HTTP-Server
Set-Cookie:
JSESSIONID=f25fe9baa0bb29c93561749bc67ea5f9f03c8b907681f2c13908aa6042d0eb88.e34Kc3
0QahiQa00Lahn0; path=/CoreService
Connection: close
Content-Type: text/xml; charset=utf-8

<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <changeStatusResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </changeStatusResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [auditRObjct](#)

## approveRObject

**Service** To see the approval results for an object.

**Usage** It inform the users whether the object has been approved by the approver, or, when an approver is approving the object on behalf of one or more user groups. After a change is routed to group of approvers, the online approval process begins. Users listed in the Workflow table for a change can approve or reject the change.

When you approve a change, the Agile system records the approval in the Workflow table. When all approvers have approved the change, the system sends an email notification to the change analyst or component engineer indicating that the change is ready to be released.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <approveRObject
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
      <request xmlns="">
        <approveRObject>
          <classIdentifier>ECO</classIdentifier>
          <objectNumber>C00034</objectNumber>
          <password>agile</password>
          <comment>Comment</comment>
          <signoffForSelf>true</signoffForSelf>
        </approveRObject>
      </request>
    </approveRObject>
  </soapenv:Body>
</soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <approveRObjectResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
      </response>
    </approveRObjectResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

**See also** [getStatus](#), [getApprovers](#)

## rejectRObjecT

**Service** To reject a routable object.

This service informs users that the routable object is rejected by an approver, or when an approver has rejected the object on behalf of one or more user groups.

**Usage** After a change is routed to group of approvers, the online approval process begins. Users listed in the Workflow table for a change can approve or reject the change.

When you approve a change, the Agile system records the approval in the Workflow table. When all approvers have approved the change, the system sends an email notification to the change analyst or component engineer indicating that the change is ready to be released.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <rejectRObjecT
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <rejectRObjecT>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00041</objectNumber>
            <password>agile</password>
            <comment>Comment</comment>
            <notifiers>
              <classIdentifier>11610</classIdentifier>
              <objectIdentifier>User11239364588798</objectIdentifier>
            </notifiers>
            <signoffForSelf>true</signoffForSelf>
          </rejectRObjecT>
        </request>
      </rejectRObjecT>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <rejectRObjecTResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </rejectRObjecTResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** getApprovers, getStatus, auditRObjecT, approveRObjecT

## setWorkflow

**Service** To set the workflow of an object.

**Usage** As long as a change is in the Pending status, you have option to set a different workflow. Once a change moves beyond Pending status, you cannot change the workflow. If a routable object has not been assigned a workflow yet, you can use the setWorkflow method to set the workflow.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <setWorkflow
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <setWorkflowRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00033</objectNumber>
            <workFlowIdentifier>3752</workFlowIdentifier>
          </setWorkflowRequest>
        </request>
      </setWorkflow>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <setWorkflowResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </setWorkflowResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getWorkFlows](#)



## addApprovers

**Service** To add a list of Approvers or Observers to a routable object.

addApprovers is used for adding a set of approvers for a given status in Agile PLM. Details of status and list of approvers can be specified in the request object. Success of the operation can be verified using the status code in the response object.

**Usage** When a change is routed and the online approval process has begun, it may be necessary to add or remove people from the list of approvers or observers. When you use addApprovers services, you specify the lists of approvers and observers, whether the notification is urgent, and an optional comment.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <addApprovers
      xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
      <request xmlns="">
        <addApproversRequest>
          <classIdentifier>ECO</classIdentifier>
          <objectNumber>C00074</objectNumber>
          <statusIdentifier>CCB</statusIdentifier>
          <approvers>
            <classIdentifier>11610</classIdentifier>
            <objectIdentifier>User11239100555679</objectIdentifier>
          </approvers>
          <approvers>
            <classIdentifier>11610</classIdentifier>
            <objectIdentifier>User21239100555679</objectIdentifier>
          </approvers>
          <urgent>false</urgent>
          <comment>Comments</comment>
        </addApproversRequest>
      </request>
    </addApprovers>
  </soapenv:Body>
</soapenv:Envelope>

==== Response ====
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <addApproversResponse
      xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
      </response>
    </addApproversResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

**See also** getWorkFlows, getStatus

## removeApprovers

**Service** Removes the approvers or observers added to a routable object.

**Usage** After a change has been routed and the online approval process has begun, it may be necessary to remove people from the list of approvers or observers. When you use removeApprovers services, you specify the lists of approvers and observers, whether the notification is urgent, and an optional comment.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeApprovers
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <removeApproversRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00042</objectNumber>
            <statusIdentifier>CCB</statusIdentifier>
            <approvers>
              <classIdentifier>11610</classIdentifier>
              <objectIdentifier>admin</objectIdentifier>
            </approvers>
            <comment>Comments</comment>
          </removeApproversRequest>
        </request>
      </removeApprovers>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeApproversResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </removeApproversResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getApprovers](#), [addApprovers](#)

## commentRObject

**Service** To comment a routable object.

**Usage** When you comment a change, you send a comment to other CCB reviewers during the online approval process. In addition to the comment, you can specify whether to notify the originator, the change analyst, and the change control board.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <commentRObject
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <commentRObjectRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00037</objectNumber>
            <comment>Comment</comment>
            <notifyOriginator>true</notifyOriginator>
            <notifyChangeAnalyst>true</notifyChangeAnalyst>
            <notifyCCB>true</notifyCCB>
            <notifyList>
              <classIdentifier>11610</classIdentifier>
              <objectIdentifier>admin</objectIdentifier>
            </notifyList>
          </commentRObjectRequest>
        </request>
      </commentRObject>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <commentRObjectResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </commentRObjectResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## getReviewers

**Service** To to obtain all the approvers for a particular change and status in the Agile system..

**Usage** The request object consists of the specifications that identify a change and status while the response object will retrieve all the approvers queried for. An array of requests of type AgileGetReviewersRequestType. This includes details about the object for whose workflow the reviewers are to be retrieved. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getReviewers
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <reviewersRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>TEST ECO2470651</objectNumber>
            <statusIdentifier>CCB</statusIdentifier>
          </reviewersRequest>
        </request>
      </getReviewers>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getReviewersResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Collaboration/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <reviewersResponse>
            <identifier>
              <classId>6141</classId>
              <className>ECO</className>
              <classDisplayName>ECO</classDisplayName>
              <objectId>6099249</objectId>
              <objectName>TEST ECO2470651</objectName>
            </identifier>
            <observers>
              <id>6099259</id>
              <name>TEST USER GROUP2 NAME1493897</name>
            </observers>
          </reviewersResponse>
        </response>
      </getReviewersResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```





## Doc Publishing Web Services

This chapter includes the following:

- loadXMLSchema ..... 193
- loadXMLData ..... 194

### loadXMLSchema

**Service** Returns an XML package that fully describes the attributes of the object.

**Usage** This is used to create XML schema files that will be used by BI Publisher to create the Templates.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadXMLSchema
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/DocPublishing/V1">
        <request xmlns="">
          <requests>
            <classIdentifiers>Part</classIdentifiers>
          </requests>
        </request>
      </loadXMLSchema>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadXMLSchemaResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/DocPublishing/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <url>http://LAP-MKAGARWA-
WF:8888/webfs/DownloadServlet?token=CCB1322A8018407FB0DDF2C10F49C3C8D5B5F4C1725C94
DA0259E3874D2D58B65F1740E2C4EFD98A13313C899D2BB3602D5030762A88DB4AE9CBDB5FD010C43E
2CFCC938C6C4B6B284806A58489D&vault=&fileID=61C735C6EBB8BDEE33</url>
          </responses>
        </response>
      </loadXMLSchemaResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## loadXMLData

**Service** Returns the actual data that is stored for an object in an XML package.

**Usage** This web service is used to retrieve the object data that is combined with the Template to create the output file. You can also use the saved output from this Web Service to test a Template in BI Publisher.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadXMLData
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/DocPublishing/V1">
        <request xmlns="">
          <requests>
            <objectReferent>
              <classIdentifier>Part</classIdentifier>
              <objectIdentifier>P0001</objectIdentifier>
            </objectReferent>
            <filterName>DefaultPartFilter</filterName>
          </requests>
        </request>
      </loadXMLData>
    </soapenv:Body>
  </soapenv:Envelope>

==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadXMLDataResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/DocPublishing/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <url>http://LAP-MKAGARWA-
WF:8888/webfs/DownloadServlet?token=753F08D64B2FF2ED9C5257A3D8EB08E12E9B669C2C2D1D
3AEFF9A733768634801513492E9A08E12EF38BCAAC38644A34EB546F4272CB08C561D9197D66B9B6DB
6B1F0127EB65E7167CECF385710&vaul=;fileID=22C1D1CB8BA62AFDB7</url>
          </responses>
        </response>
      </loadXMLDataResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```







## Folder Web Services

This chapter includes the following:

▪ getFolder.....	197
▪ createFolder.....	199
▪ renameFolder.....	200
▪ deleteFolder.....	201
▪ addChildNode.....	202
▪ getChildNode.....	203
▪ removeChildNode.....	204

### getFolder

**Service** To load folders.

**Usage** The operation retrieves the user's specific folders, such as Home Folder, Recent Visits, Favorites etc. The operation can be used by the UI clients to provide folder browsing capability. The operation can also be used to get the logged-in user's folders.

#### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getFolder xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <request xmlns="">
          <requests>
            <folderIdentifier>
              <baseFolder>3</baseFolder>
              <name>Personal Searches/TestFOLDERA3765379</name>
            </folderIdentifier>
            <recursive>true</recursive>
          </requests>
        </request>
      </getFolder>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getFolderResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folder>
              <identifier>
                <baseFolder>3</baseFolder>
                <name>Personal Searches/TestFOLDERA3765379</name>
              </identifier>
            </folder>
          </responses>
        </response>
      </getFolderResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

```
        </identifier>
        <folderType>0</folderType>
        <owner>admin</owner>
    </folder>
</responses>
</response>
</getFolderResponse>
</soapenv:Body>
</soapenv:Envelope>
```

## createFolder

**Service** To create folders.

**Usage** The operation is used to create personal folders and child folders. The operation also provides ability to set the access rights from default private to public.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createFolder
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <request xmlns="">
          <requests>
            <folderIdentifier>
              <baseFolder>3</baseFolder>
              <name>Personal Searches/TestFOLDERA4472326</name>
            </folderIdentifier>
          </requests>
        </request>
      </createFolder>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createFolderResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folder>
              <identifier>
                <baseFolder>3</baseFolder>
                <name>Personal Searches/TestFOLDERA4472326</name>
                <version>1</version>
              </identifier>
              <folderType>0</folderType>
              <owner>admin</owner>
            </folder>
          </responses>
        </response>
      </createFolderResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

## renameFolder

**Service** To rename a folder.

**Usage** The operation passes the value of original folder name and the new name for it.

**Sample Code SOAP**

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <renameFolder
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <request xmlns="">
          <requests>
            <folderIdentifier>
              <baseFolder>3</baseFolder>
              <name>Personal Searches/TESTFOLDER1517677</name>
            </folderIdentifier>
            <newName>newTestFOLDER8320018</newName>
          </requests>
        </request>
      </renameFolder>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <renameFolderResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folder>
              <identifier>
                <baseFolder>3</baseFolder>
                <name>Personal Searches/newTestFOLDER8320018</name>
                <version>2</version>
              </identifier>
              <folderType>0</folderType>
              <owner>admin</owner>
            </folder>
          </responses>
        </response>
      </renameFolderResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## deleteFolder

**Service** To delete a folder.

**Usage** Given the folder identifier, the operation deletes that folder.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteFolder
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <request xmlns="">
          <requests>
            <folderIdentifier>
              <baseFolder>3</baseFolder>
              <name>Personal Searches/TestFOLDERA3765379</name>
            </folderIdentifier>
            <recursive>false</recursive>
          </requests>
        </request>
      </deleteFolder>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteFolderResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </deleteFolderResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## addChildNode

**Service** To add child Nodes to a folder.

**Usage** The list of child Nodes to be added to a folder are passed as input to the request. The child nodes can be a query object, an AgileObject or a subfolder itself.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addChildNode
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <request xmlns="">
          <requests>
            <folderIdentifier>
              <baseFolder>26</baseFolder>
              <name>TEST ITEMS</name>
            </folderIdentifier>
            <agileObjects>
              <classIdentifier>Part</classIdentifier>
              <objectIdentifier>TEST PART16202692</objectIdentifier>
            </agileObjects>
          </requests>
        </request>
      </addChildNode>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addChildNodeResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </addChildNodeResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```



## getChildNode

**Service** To get child Nodes of a folder.

**Usage** The list of child Nodes to be returned for a folder is passed as input to the request. The returned childNode can be a query object, an AgileObject or a subfolder itself.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getChildNode
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <request xmlns="">
          <requests>
            <folderIdentifier>
              <baseFolder>26</baseFolder>
              <name>TESTFOLDERA5703774</name>
            </folderIdentifier>
            <children>TESTFOLDERB8643248</children>
            <allChildren>>false</allChildren>
          </requests>
        </request>
      </getChildNode>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getChildNodeResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <folderIdentifier>
              <baseFolder>26</baseFolder>
              <name>TESTFOLDERA5703774</name>
              <version>3</version>
            </folderIdentifier>
            <folders>
              <baseFolder>26</baseFolder>
              <name>TESTFOLDERA5703774/TESTFOLDERB8643248</name>
              <version>4</version>
            </folders>
          </responses>
        </response>
      </getChildNodeResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## removeChildNode

**Service** To remove child Nodes from a folder.

**Usage** The list of child Nodes to be removed is passed as input to the request. You can also choose to remove all child Nodes of a folder by setting a flag present on the request Object. The child Node can be a query object, an AgileObject or a subfolder itself.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeChildNode
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <request xmlns="">
          <requests>
            <folderIdentifier>
              <baseFolder>26</baseFolder>
              <name>TESTFOLDERA1611270</name>
            </folderIdentifier>
            <children>TESTFOLDERB5695801</children>
            <allChildren>>false</allChildren>
          </requests>
        </request>
      </removeChildNode>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeChildNodeResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Folder/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </removeChildNodeResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```





## PC Web Services

This chapter includes the following:

▪ setIncorporate.....	207
▪ getRevisions .....	208
▪ undoRedline.....	210
▪ isRedlineModified .....	211
▪ isIncorporated.....	212

### setIncorporate

**Service** To set the status of an Agile object as 'incorporated' or 'unincorporated'.

**Usage** The request object is formed based on the class and object identifiers of the object and the status of incorporation. Success of the operation is verified using the status code in the response object.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <setIncorporate xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
      <request xmlns="">
        <requests>
          <classIdentifier>Part</classIdentifier>
          <objectNumber>P00735</objectNumber>
          <incorporate>true</incorporate>
        </requests>
      </request>
    </setIncorporate>
  </soapenv:Body>
</soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <setIncorporateResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
        <responses>
          <isIncorporated>true</isIncorporated>
        </responses>
      </response>
    </setIncorporateResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

## getRevisions

- Service** To retrieve the revisions of an Agile object given the details of the object and relevant options.
- Usage** The request object is formed based this information and revisions of the object are obtained through the response object. Success of the operation is verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getRevisions xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>1000-02</objectNumber>
            <allRevisions>false</allRevisions>
          </requests>
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>1000-02</objectNumber>
            <allRevisions>true</allRevisions>
          </requests>
        </request>
      </getRevisions>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getRevisionsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <currentRev>C</currentRev>
          </responses>
          <responses>
            <currentRev>C</currentRev>
            <revisions>
              <changeIdentifier>
                <classId>6141</classId>
                <className>ECO</className>
                <classDisplayName>ECO</classDisplayName>
                <objectId>45</objectId>
                <objectName>25000</objectName>
              </changeIdentifier>
              <revision>(D)</revision>
            </revisions>
            <revisions>
              <changeIdentifier>
                <classId>6141</classId>
                <className>ECO</className>
                <classDisplayName>ECO</classDisplayName>
                <objectId>44</objectId>
                <objectName>24433</objectName>
              </changeIdentifier>
              <revision>C</revision>
            </revisions>
          </responses>
        </response>
      </getRevisionsResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

```

      <changeIdentifier>
        <classId>6141</classId>
        <className>ECO</className>
        <classDisplayName>ECO</classDisplayName>
        <objectId>43</objectId>
        <objectName>24020</objectName>
      </changeIdentifier>
      <revision>B</revision>
    </revisions>
  <revisions>
    <changeIdentifier>
      <classId>6141</classId>
      <className>ECO</className>
      <classDisplayName>ECO</classDisplayName>
      <objectId>41</objectId>
      <objectName>23450</objectName>
    </changeIdentifier>
    <revision>A</revision>
  </revisions>
  <revisions>
    <changeIdentifier xsi:nil="true"/>
    <revision>Introductory</revision>
  </revisions>
</responses>
</response>
</getRevisionsResponse>
</soapenv:Body>
</soapenv:Envelope>

```

**See also**      getObject, loadTable

## undoRedline

**Service** To revert a redlined entity in Agile PLM by issuing an undo operation on the redline.

**Usage** Relevant details are used to form the request object. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <undoRedline xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P1242809159264</objectNumber>
            <redlineTable>TABLE REDLINEBOM</redlineTable>
            <rowId>6201465</rowId>
            <options>
              <propertyName>revision</propertyName>
              <propertyValue>B</propertyValue>
            </options>
          </requests>
        </request>
      </undoRedline>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <undoRedlineResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses/>
        </response>
      </undoRedlineResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** loadTable, getRowId Method on page 285



## isRedlineModified

**Service** To determine whether a particular redlined entity in Agile PLM has been modified or not.

**Usage** Relevant details are used to form the request object. The response object includes information that will denote whether the specified red line was modified.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <isRedlineModified
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <request xmlns="">
          <requests>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>P1242818906603</objectNumber>
            <redlineTable>TABLE REDLINEBOM</redlineTable>
            <rowId>6201729</rowId>
            <options>
              <propertyName>revision</propertyName>
              <propertyValue>B</propertyValue>
            </options>
          </requests>
        </request>
        <requests>
          <classIdentifier>Part</classIdentifier>
          <objectNumber>P1242818906603</objectNumber>
          <redlineTable>TABLE REDLINEBOM</redlineTable>
          <rowId>6201734</rowId>
          <options>
            <propertyName>revision</propertyName>
            <propertyValue>B</propertyValue>
          </options>
        </requests>
      </isRedlineModified>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <isRedlineModifiedResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <isRedlineModified>true</isRedlineModified>
          </responses>
          <responses>
            <isRedlineModified>false</isRedlineModified>
          </responses>
        </response>
      </isRedlineModifiedResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

**See also** loadTable, getRowId Method on page 285

## isIncorporated

<b>Service</b>	To determine whether a particular data object entity in Agile PLM has been Incorporated.
<b>Usage</b>	Details of the object whose 'Is Incorporated' status is to be retrieved are specified in an array of requests. The response object includes information that will denote whether the specified data object has been Incorporated or not.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <isIncorporated xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <requests>
            <classIdentifier>10141</classIdentifier>
            <objectNumber>TEST PART5780230</objectNumber>
          </requests>
        </request>
      </isIncorporated>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <isIncorporatedResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Pc/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <isIncorporated>false</isIncorporated>
          </responses>
        </response>
      </isIncorporatedResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```





## Project Web Services

This chapter includes the following:

▪ createBaseline .....	215
▪ removeBaseline .....	217
▪ getBaselines .....	218
▪ delegateOwnership .....	219
▪ substituteResource .....	220
▪ isLocked .....	221
▪ setLock .....	222
▪ reschedule .....	223
▪ assignUsersFromPool .....	224

### createBaseline

**Service** Creates a new Project baseline.

**Usage** It contains information about the Project Object and the name by which the new baseline needs to be created. You can save multiple baselines, and retrieve them later for comparison. The operation is used to create a baseline (snapshot) for the program object. The baseline name will be sent as an input to the request..

Baselines can be created only for the root program object.

#### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createBaseline
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
        <request xmlns="">
          <requests>
            <projectIdentifier>PGM00007</projectIdentifier>
            <description>TEST BASELINE</description>
            <kickoffBaseline>true</kickoffBaseline>
          </requests>
        </request>
      </createBaseline>
    </soapenv:Body>
  </soapenv:Envelope>
  <?xml version="1.0" encoding="utf-8"?>
    <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <soapenv:Body>
        <createBaselineResponse
          xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
          <response xmlns="">
            <messageId xsi:nil="true"/>
            <messageName xsi:nil="true"/>
            <statusCode>SUCCESS</statusCode>
            <responses>
              <version>2</version>
            </responses>
          </response>
        </createBaselineResponse>
      </soapenv:Body>
    </soapenv:Envelope>
```

```
<classIdentifier>Program</classIdentifier>
<objectIdentifier>PGM00007</objectIdentifier>
<version>0</version>
</project>
</responses>
</response>
</createBaselineResponse>
</soapenv:Body>
</soapenv:Envelope>
```

## removeBaseline

**Service** Removes the list of all baselines associated with a program object.

**Usage** The request object should have the ClassIdentifier and the ObjectIdentifier of the Program Object on which the baselines are being removed. It should also contain the name of the baselines being removed.

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <removeBaseline
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <request xmlns="">
        <requests>
          <projectIdentifier>PGM00011</projectIdentifier>
          <version>1</version>
        </requests>
      </request>
    </removeBaseline>
  </soapenv:Body>
</soapenv:Envelope>
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <removeBaselineResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
        <responses>
          <project>
            <classIdentifier>Program</classIdentifier>
            <objectIdentifier>PGM00011</objectIdentifier>
            <version>0</version>
          </project>
        </responses>
      </response>
    </removeBaselineResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

## getBaselines

**Service** Retrieves the list of all baselines associated with a program object.

**Usage** The getBaselines operation selects a Project baseline. The request object should have the ClassIdentifier and the ObjectIdentifier of the Program Object for which the baselines are being loaded.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getBaselines
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
        <request xmlns="">
          <requests>
            <projectIdentifier>PGM00009</projectIdentifier>
          </requests>
        </request>
      </getBaselines>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getBaselinesResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <baselines>
              <version>1</version>
              <description>TEST BASELINE</description>
              <milestone xsi:nil="true"/>
              <kickoffBaseline>false</kickoffBaseline>
              <planOfRecord>true</planOfRecord>
              <createDate>2010-05-21T08:48:52.000Z</createDate>
            </baselines>
            <baselines>
              <version>2</version>
              <description>TEST BASELINE2</description>
              <milestone xsi:nil="true"/>
              <kickoffBaseline>false</kickoffBaseline>
              <planOfRecord>false</planOfRecord>
              <createDate>2010-05-21T08:48:52.000Z</createDate>
            </baselines>
            <project>
              <classIdentifier>Program</classIdentifier>
              <objectIdentifier>PGM00009</objectIdentifier>
              <version>0</version>
            </project>
          </responses>
        </response>
      </getBaselinesResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```



## delegateOwnership

- Service** To let the owner or program manager of a program object to assign the ownership of the program to other users by delegating it.
- Usage** The delegated user receives a request which may be accepted or rejected. If the delegation is accepted, the delegated user becomes the owner of the task. A delegated owner is automatically given the Program Manager role for the delegated program object.
- The web service request specifies the user to whom the ownership must be delegated and whether the act of delegation also applies to the program's children.

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <delegateOwnership
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <request xmlns="">
        <requests>
          <projectIdentifier>PGM00008</projectIdentifier>
          <userIdentifier>yvonnec</userIdentifier>
          <applyToChildren>true</applyToChildren>
        </requests>
      </request>
    </delegateOwnership>
  </soapenv:Body>
</soapenv:Envelope>
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <delegateOwnershipResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
        <responses>
          <project>
            <classIdentifier>Program</classIdentifier>
            <objectIdentifier>PGM00008</objectIdentifier>
            <version>2</version>
          </project>
        </responses>
      </response>
    </delegateOwnershipResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

## substituteResource

**Service** To enable the substitution of one resource, a user or a user group, with another.

**Usage** For a given program, the current resource's role may be assigned to the substituted resource. When resources are to be substituted, users as well as user groups may be used. The request will also specify whether the substitution of resources is propagated to the the program's children..

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <substituteResource
      xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <request xmlns="">
        <requests>
          <projectIdentifier>PGM00014</projectIdentifier>
          <currentResource>
            <classIdentifier>11610</classIdentifier>
            <objectIdentifier>badriv</objectIdentifier>
          </currentResource>
          <newResource>
            <classIdentifier>11610</classIdentifier>
            <objectIdentifier>yvonnec</objectIdentifier>
          </newResource>
          <applyToChildren>true</applyToChildren>
        </requests>
      </request>
    </substituteResource>
  </soapenv:Body>
</soapenv:Envelope>
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <substituteResourceResponse
      xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
        <responses>
          <project>
            <classIdentifier>Program</classIdentifier>
            <objectIdentifier>PGM00014</objectIdentifier>
            <version>6</version>
          </project>
        </responses>
      </response>
    </substituteResourceResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

## isLocked

**Service** To check the locked status of a program object

**Usage** The operation determine whether a Program object in Agile is locked. The request object should have the ClassIdentifier and the ObjectIdentifier of the Program Object that is being checked.

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <isLocked xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <request xmlns="">
        <requests>
          <projectIdentifier>PGM00010</projectIdentifier>
        </requests>
      </request>
    </isLocked>
  </soapenv:Body>
</soapenv:Envelope>
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <isLockedResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
        <responses>
          <isLocked>false</isLocked>
          <project>
            <classIdentifier>Program</classIdentifier>
            <objectIdentifier>PGM00010</objectIdentifier>
            <version>0</version>
          </project>
        </responses>
      </response>
    </isLockedResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

## setLock

**Service** To lock or unlock a program object.

**Usage** The request object should have the ClassIdentifier and the ObjectIdentifier of the Program Object that is being locked..

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <setLock xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
        <request xmlns="">
          <requests>
            <projectIdentifier>PGM00013</projectIdentifier>
            <lock>false</lock>
          </requests>
        </request>
      </setLock>
    </soapenv:Body>
  </soapenv:Envelope>
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <setLockResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <project>
              <classIdentifier>Program</classIdentifier>
              <objectIdentifier>PGM00013</objectIdentifier>
              <version>0</version>
            </project>
          </responses>
        </response>
      </setLockResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## reschedule

**Service** To reschedule a Program object.

**Usage** Once a program's schedule has been defined, use this operation to reschedule it. The operation will enable the client to change the startDate and endDate, and move the program backward or forward.

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <reschedule xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <request xmlns="">
        <requests>
          <projectIdentifier>PGM00012</projectIdentifier>
          <rescheduleType>
            <startDate>2009-08-14T18:30:00.000Z</startDate>
          </rescheduleType>
        </requests>
      </request>
    </reschedule>
  </soapenv:Body>
</soapenv:Envelope>
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <rescheduleResponse
      xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
        <responses>
          <project>
            <classIdentifier>Program</classIdentifier>
            <objectIdentifier>PGM00012</objectIdentifier>
            <version>2</version>
          </project>
        </responses>
      </response>
    </rescheduleResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

## assignUsersFromPool

**Service** To assign users from a user group.

**Usage** In Agile Web Client, when you add a resource pool to the Team table, you can replace the pool with one or more resources contained within it. That is, instead of assigning the entire resource pool, you can assign select users from the pool. The operation reproduces this functionality. The operation makes detailed resource assignments by selecting a specific user from a group used as team member.

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <assignUsersFromPool
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
        <request xmlns="">
          <requests>
            <projectIdentifier>PGM00006</projectIdentifier>
            <users>demol</users>
            <usergroup>TEST USER GROUP NAME747945</usergroup>
          </requests>
        </request>
      </assignUsersFromPool>
    </soapenv:Body>
  </soapenv:Envelope>
  <?xml version="1.0" encoding="utf-8"?>
    <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <soapenv:Body>
        <assignUsersFromPoolResponse
          xmlns="http://xmlns.oracle.com/AgileObjects/Core/Project/V1">
          <response xmlns="">
            <messageId xsi:nil="true"/>
            <messageName xsi:nil="true"/>
            <statusCode>SUCCESS</statusCode>
            <responses>
              <project>
                <classIdentifier>Program</classIdentifier>
                <objectIdentifier>PGM00006</objectIdentifier>
                <version>4</version>
              </project>
            </responses>
          </response>
        </assignUsersFromPoolResponse>
      </soapenv:Body>
    </soapenv:Envelope>
```







## Recipe & Material Workspace Web Services

This chapter includes the following:

▪ search .....	227
▪ edit .....	229
▪ processBO .....	230

### search

**Service** A generic search Web service.

**Usage** Perform search operations on any category by providing the search criteria in cfmXML format.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
<cfmXML>
  <LoginInfo>
    <UserName>admin</UserName>
    <Password>ANrE.7E.CNYiKbvzxDrqFJGtUJ.Olk</Password>
  </LoginInfo>
  <SessionInfo logout="true">
    <UserName>admin</UserName>
    <SessionID/>
  </SessionInfo>
  <SearchCriteria id="1">
    <CategoryName>User</CategoryName>
    <CategoryDBName>D PPL USER</CategoryDBName>
    <ViewName/>
    <Attribute name="User ID">
      <Value dataType="Text">Administrator</Value>
      <UnitOfMeasure/>
    </Attribute>
  </SearchCriteria>
</cfmXML>
==== Response ====
<?xml version="1.0" encoding="UTF-8"?>
<cfmXML>
  <Response>
    <ResponseCode>500</ResponseCode>
    <ResponseMessage>Search Successfully</ResponseMessage>
    <SessionID>7676009502140705176</SessionID>
  </Response>
  <Payload>
    <Object>
      <CategoryName>User</CategoryName>
      <CategoryDBName>D PPL USER</CategoryDBName>
      <ViewName>DEFAULT</ViewName>
      <ObjectKey>
        <Attribute name="User ID">
          <Value dataType="Text">admin</Value>
          <UnitOfMeasure/>
        </Attribute>
      </ObjectKey>
      <ObjectDetail>
        <Attribute name="Internal Id">
          <Value dataType="Number">2701</Value>
          <UnitOfMeasure/>
        </Attribute>
      </ObjectDetail>
    </Object>
  </Payload>
</cfmXML>
```

```

PM</Value>
    <Attribute name="Effective Date">
      <Value dataType="DateTime">09/25/2005 10:54:44
    </Value>
      <UnitOfMeasure/>
    </Attribute>
    <Attribute name="Effective End Date">
      <Value dataType="DateTime"/>
      <UnitOfMeasure/>
    </Attribute>
    <Attribute name="Is Active">
      <Value dataType="Boolean">true</Value>
      <UnitOfMeasure/>
    </Attribute>
    <Attribute name="Modified Date">
      <Value dataType="DateTime">09/27/2005 04:05:20
PM</Value>
      <UnitOfMeasure/>
    </Attribute>
    <Relationship name="Created User">
      <CategoryName>User</CategoryName>
      <CategoryDBName>D PPL USER</CategoryDBName>
      <ObjectKey>
        <Attribute name="User ID">
          <Value
dataType="Text">admin</Value>
            <UnitOfMeasure/>
          </Attribute>
        </ObjectKey>
      </Relationship>
      <Relationship name="Modified User">
        <CategoryName>User</CategoryName>
        <CategoryDBName>D PPL USER</CategoryDBName>
        <ObjectKey>
          <Attribute name="User ID">
            <Value
dataType="Text">application</Value>
              <UnitOfMeasure/>
            </Attribute>
          </ObjectKey>
        </Relationship>
      </ObjectDetail>
    </Object>
  </Payload>
</cfmXML>

```

## edit

**Service** A generic edit web service.

**Usage** Perform edit operation on any category by providing the payload value in cfmXML format.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
<cfmXML>
<LoginInfo>
  <UserName>admin</UserName>
  <Password>ANrE.7E.CNYiKbvzxDrqFJGtUJ.Olk</Password>
</LoginInfo>
<Payload>
  <Object operation="I">
    <CategoryName>Application Setup</CategoryName>
    <CategoryDBName>D INT APPLICATION SETUP</CategoryDBName>
    <ViewName>DEFAULT</ViewName>
    <ObjectKey>
      <Attribute name="Application Name">
        <Value dataType="Basic Text">Test App Name</Value>
        <UnitOfMeasure/>
      </Attribute>
    </ObjectKey>
    <ObjectDetail>
      <Attribute name="Application User Id">
        <Value dataType="Basic Text">Test App User
Id</Value>
        <UnitOfMeasure/>
      </Attribute>
      <Attribute name="Inbound XSL">
        <Value dataType="Basic Text">Test Inbound
XSL</Value>
        <UnitOfMeasure/>
      </Attribute>
      <Attribute name="Outbound XSL">
        <Value dataType="Basic Text">Test Outbound
XSL</Value>
        <UnitOfMeasure/>
      </Attribute>
    </ObjectDetail>
  </Object>
</Payload>
</cfmXML>

==== Response ====
<?xml version="1.0" encoding="UTF-8"?>
<cfmXML>
<Response>
  <ResponseCode>500</ResponseCode>
  <ResponseMessage>Edit operation completed successfully.</ResponseMessage>
  <SessionID>8360097756062912387</SessionID>
</Response>
<Payload>
  <Object operation="I">
    <CategoryName>Application Setup</CategoryName>
    <CategoryDBName>D INT APPLICATION SETUP</CategoryDBName>
    <ViewName>DEFAULT</ViewName>
    <ObjectKey>
      <Attribute name="Application Name">
        <Value dataType="Basic Text">Test App Name</Value>
        <UnitOfMeasure/>
      </Attribute>
    </ObjectKey>
    <ObjectDetail>
      <Attribute name="NoteCount">
        <Value dataType="Integer">0</Value>
        <UnitOfMeasure/>
      </Attribute>
    </ObjectDetail>
  </Object>
</Payload>
</cfmXML>

```

```

        </Attribute>
        <Attribute name="AttachmentCount">
          <Value dataType="Integer">0</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Internal Id">
          <Value dataType="Integer">873627</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="LeafCategory Id">
          <Value dataType="Foreign Key">@</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Application User Id">
          <Value dataType="Basic Text">Test App User
Id</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Application User Password">
          <Value dataType="Password">@</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Inbound XSL">
          <Value dataType="Basic Text">Test Inbound
XSL</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Outbound XSL">
          <Value dataType="Basic Text">Test Outbound
XSL</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Max Payload objects">
          <Value dataType="Integer">@</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Created Date">
          <Value GMTDateTime="1165838083000"
dataType="DateTime">11/12/2006 17:24:43 IST</Value>
          <UnitOfMeasure/>
          <Format>dd/MM/yyyy HH:mm:ss z</Format>
        </Attribute>
        <Attribute name="Modified Date">
          <Value GMTDateTime="1165838083000"
dataType="DateTime">11/12/2006 17:24:43 IST</Value>
          <UnitOfMeasure/>
          <Format>dd/MM/yyyy HH:mm:ss z</Format>
        </Attribute>
        <Relationship name="Created User">
          <CategoryName>User</CategoryName>
          <CategoryDBName>D PPL USER</CategoryDBName>
          <ObjectKey>
            <ObjectID>785544</ObjectID>
            <Attribute name="User ID">
              <Value dataType="Basic
Text">admin</Value>
              <UnitOfMeasure/>
            </Attribute>
          </ObjectKey>
        </Relationship>
        <Relationship name="Modified User">
          <CategoryName>User</CategoryName>
          <CategoryDBName>D PPL USER</CategoryDBName>
          <ObjectKey>
            <ObjectID>785544</ObjectID>
            <Attribute name="User ID">
              <Value dataType="Basic
Text">admin</Value>
              <UnitOfMeasure/>
            </Attribute>
          </ObjectKey>
        </Relationship>
      </ObjectDetail>
    </Object>
  </Payload>
</cfmXML>

```

## processBO

**Service** A generic web service to process any Business Objects.

**Usage** Operations such as Add/Modify/Delete can be performed on Business Object by just invoking this web service with BO details in the form of cfmXML.

### Sample Code SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<cfmXML>
<LoginInfo>
  <UserName>admin</UserName>
  <Password>ANrE.7E.CNYiKbvzxDrqFJGtUJ.Olk</Password>
</LoginInfo>
<Payload>
  <Object>
    <CategoryName/>
    <CategoryDBName/>
    <ObjectKey/>
    <ObjectDetail/>
  </Object>
  <ObjectGroup isBO="Yes" name="Application">
    <BOActions>
      <BOAction>New</BOAction>
    </BOActions>
    <Object name="BORoot">
      <CategoryName>Application Setup</CategoryName>
      <CategoryDBName>D INT APPLICATION SETUP</CategoryDBName>
      <ViewName/>
      <CustomInfo/>
      <ObjectKey>
        <Attribute name="Application Name">
          <Value
dataType="Text">JUNIT BO WS Test Application 041</Value>
          <UnitOfMeasure/>
        </Attribute>
      </ObjectKey>
      <ObjectDetail>
        <Attribute name="Inbound XSL">
          <Value dataType="Text">inbound</Value>
          <UnitOfMeasure/>
        </Attribute>
        <Attribute name="Outbound XSL">
          <Value dataType="Text">outbound</Value>
          <UnitOfMeasure/>
        </Attribute>
      </ObjectDetail>
    </Object>
  </ObjectGroup>
  <ObjectGroup isBO="Yes" name="Application">
    <BOActions>
      <BOAction>New</BOAction>
    </BOActions>
    <Object name="BORoot">
      <CategoryName>Application Setup</CategoryName>
      <CategoryDBName>D INT APPLICATION SETUP</CategoryDBName>
      <ViewName/>
      <CustomInfo/>
      <ObjectKey>
        <Attribute name="Application Name">
          <Value
dataType="Text">JUNIT BO WS Test Application 002</Value>
          <UnitOfMeasure/>
        </Attribute>
      </ObjectKey>
      <ObjectDetail>
        <Attribute name="Inbound XSL">
          <Value dataType="Text">inbound</Value>
          <UnitOfMeasure/>
        </Attribute>
      </ObjectDetail>
    </Object>
  </ObjectGroup>
</Payload>
</cfmXML>
```

```
        </Attribute>
        <Attribute name="Outbound XSL">
            <Value dataType="Text">outbound</Value>
            <UnitOfMeasure/>
        </Attribute>
    </ObjectDetail>
</Object>
</ObjectGroup>
</Payload>
</cfmXML>
```

## Reports Web Services

This chapter includes the following:

- loadReportSchema..... 233
- loadReportData..... 234

### loadReportSchema

**Service** To load the report schema.

**Usage** The .

**Sample Code SOAP**

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadReportSchema
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Report/V1">
        <request xmlns="">
          <requests>
            <reportIdentifier>Project Assembly Cost
Report</reportIdentifier>
            <params>
              <id>objectNumber</id>
              <values>PRJ00001</values>
            </params>
          </requests>
        </request>
      </loadReportSchema>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadReportSchemaResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Report/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <reportIdentifier>103</reportIdentifier>
            <url>http://MMUTHUSW-
IDC:8888/webfs/DownloadServlet?token=A58AB33627AC4B6033077D4A63FF204E335E3378E5224
91A3CF50275A8D78F029DC2D464BB66C560E5F726C6E2044EE9C28BE814D82089565C9A16DEDAD793C
6F80275F5D17673C20369CEA74EDF&vault=&fileID=C861DB6AFABCEB27FD</url>
          </responses>
        </response>
      </loadReportSchemaResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## loadReportData

**Service** To load data of a report.

**Usage** .

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadReportData
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Report/V1">
        <request xmlns="">
          <requests>
            <reportIdentifier>Project Assembly Cost
Report</reportIdentifier>
            <params>
              <id>objectNumber</id>
              <values>PRJ00001</values>
            </params>
            <params>
              <id>items</id>
              <values>CAP500-PS</values>
              <values>CAP450-PS</values>
            </params>
            <params>
              <id>suppliers</id>
              <values>EMS1 COMPONENT SUPPLIER</values>
              <values>EMS2 COMPONENT MFR</values>
            </params>
            <params>
              <id>pricepoints</id>
              <values>04/01/2004 To 06/30/2004 QuantityBreak1</values>
              <values>07/01/2003 To 09/29/2003 QuantityBreak1</values>
              <values>09/30/2003 To 12/29/2003 QuantityBreak1</values>
              <values>12/30/2003 To 03/31/2004 QuantityBreak1</values>
            </params>
            <params>
              <id>doCostRollup</id>
              <values>>false</values>
            </params>
          </requests>
        </request>
      </loadReportData>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
```







## Search Web Services

This chapter includes the following:

▪ quickSearch .....	237
▪ advancedSearch .....	239
▪ getSearchableAttributes .....	241
▪ getSearchableClasses .....	243
▪ createQuery .....	244
▪ loadQuery .....	246
▪ executeSavedQuery .....	248
▪ saveAsQuery .....	249
▪ updateQuery .....	251
▪ deleteQuery .....	253

### quickSearch

**Service** To retrieve a list of all Agile objects whose specifications match the search criteria specified in the request object.

**Usage** Object name, number (ID), or description may be used to form the criteria. The response object contains a collection of which of Agile objects, which were successfully queried for. Success of the operation is verified by using the status code in the response object.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <quickSearch xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <classIdentifier>Part</classIdentifier>
          <keywords>P0*</keywords>
          <searchFiles>false</searchFiles>
        </request>
      </quickSearch>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <quickSearchResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <table>
            <tableIdentifier>
              <objectId>0</objectId>
            </tableIdentifier>
          </table>
        </response>
      </quickSearchResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

```
        <tableName xsi:nil="true"/>
      </tableIdentifier>
      <row rowId="13">
        <objectReferentId>
          <classId>10141</classId>
          <className>Part</className>
          <objectId>6098830</objectId>
          <objectName xsi:nil="true"/>
        </objectReferentId>
        <ItemType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1">Part</ItemType>
        <Number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">P00003</Number>
        <Description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3"></Description>
        <LifecyclePhase xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
attributeId="4">Preliminary</LifecyclePhase>
        <Rev xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="5"></Rev>
      </row>
      <row rowId="14">
        <objectReferentId>
          <classId>10141</classId>
          <className>Part</className>
          <objectId>6098836</objectId>
          <objectName xsi:nil="true"/>
        </objectReferentId>
        <ItemType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1">Part</ItemType>
        <Number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">P00005</Number>
        <Description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3"></Description>
        <LifecyclePhase xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
attributeId="4">Preliminary</LifecyclePhase>
        <Rev xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="5"></Rev>
      </row>
    </table>
  </response>
</quickSearchResponse>
```

**See also**      [getSearchableAttributes](#), [advancedSearch](#)

## advancedSearch

- Service** To retrieve a list of all Agile objects whose specifications match the advanced search criteria specified in the request object.
- Usage** Advanced search provides options for forming complex search criteria. The response object contains a collection of which of Agile objects which were successfully queried for. Success of the operation is verified using the status code in the response object.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <advancedSearch
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <classIdentifier>Part</classIdentifier>
          <criteria>[Title Block.Number] contains 'P0' & & [Title
Block.Description] is not null</criteria>
          <caseSensitive>>false</caseSensitive>
          <displayName>Search123</displayName>
          <resultAttributes>Title Block.Number</resultAttributes>
          <resultAttributes>Title Block.Description</resultAttributes>
          <resultAttributes>Title Block.Lifecycle Phase</resultAttributes>
        </request>
      </advancedSearch>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <advancedSearchResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <table>
            <tableIdentifier>
              <objectId>6112848</objectId>
              <objectName>Search123</objectName>
              <tableId>-102</tableId>
              <tableName xsi:nil="true"/>
            </tableIdentifier>
            <row rowId="1">
              <objectReferentId>
                <classId>10141</classId>
                <className>Part</className>
                <objectId>6098826</objectId>
                <objectName xsi:nil="true"/>
              </objectReferentId>
              <number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1001">P00001</number>
              <description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1002">Gimme all yer
money, AAAARRRRR ye land-lover!!</description>
              <lifecyclePhase xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1"
attributeId="1084">
                <selection>
                  <id>976</id>
                  <apiName>PRELIMINARY</apiName>
                </selection>
              </lifecyclePhase>
            </row>
          </table>
        </response>
      </advancedSearchResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

```
        </selection>
      </lifecyclePhase>
    </row>
  </table>
</response>
</advancedSearchResponse>
```

**See also**     [getAllClasses](#), [getSearchableAttributes](#)

## getSearchableAttributes

**Service** To retrieve a list of all searchable attributes on a baseclass, class or a subclass.

**Usage** The request object is formed using relevant details. The response object contains the attributes that were queried for. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSearchableAttributes
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <classIdentifier>Part</classIdentifier>
        </request>
      </getSearchableAttributes>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSearchableAttributesResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <attributes>
            <nodeId>2000004143</nodeId>
            <apiName>complianceCalculatedDate</apiName>
            <typeATTRIBUTE</type>
            <displayName>Compliance Calculated Date</displayName>
            <dataType>3</dataType>
            <searchable>true</searchable>
            <visible>true</visible>
            <required>false</required>
            <maxLength>2147483647</maxLength>
            <properties>
              <propertyId>1</propertyId>
              <apiName>AttType</apiName>
              <displayName>AttType</displayName>
              <readOnly>false</readOnly>
              <AttType xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema"></AttType>
            </properties>
            <properties>
              <propertyId>9</propertyId>
              <apiName>Visible</apiName>
              <displayName>Visible</displayName>
              <readOnly>true</readOnly>
              <Visible xsi:type="common:AgileListEntryType"
                xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
                <selection>
                  <id>1</id>
                  <apiName>YES</apiName>
                  <value>Yes</value>
                </selection>
              </Visible>
            </properties>
          </attributes>
        </response>
      </getSearchableAttributesResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

**See also**      [getAttributes](#)



## getSearchableClasses

**Service** To retrieve a list of searchable base classes, classes and sub-classes.

**Usage** The operation allows the user to retrieve the list of Searchable BaseClasses/ Classes/ SubClasses for which the user has privilege. The request object takes classIdentifier as input. If the classIdentifier is not set then the response will contain the list of all searchable BaseClasses else the response will return the list of all subclasses of that classIdentifier.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSearchableClasses
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <onlyBaseClasses>false</onlyBaseClasses>
          <classIdentifier>ChangesBaseClass</classIdentifier>
        </request>
      </getSearchableClasses>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getSearchableClassesResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <classes>
            <nodeId>6000</nodeId>
            <apiName>ChangeOrdersClass</apiName>
            <typeCLASS</type>
            <displayName>Change Orders</displayName>
            <abstractClass>true</abstractClass>
          </classes>
          <classes>
            <nodeId>1450</nodeId>
            <apiName>ManufacturerOrdersClass</apiName>
            <typeCLASS</type>
            <displayName>Manufacturer Orders</displayName>
            <abstractClass>true</abstractClass>
          </classes>
          <classes>
            <nodeId>11000</nodeId>
            <apiName>StopShipsClass</apiName>
            <typeCLASS</type>
            <displayName>Stop Ships</displayName>
            <abstractClass>true</abstractClass>
          </classes>
        </response>
      </getSearchableClassesResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## createQuery

**Service** To create a search query and save the query into a folder.

**Usage** The createQuery web service operation creates a query given the criteria and folder path where it is to be saved. The response object contains the QueryObjectType objects. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createQuery xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <requests>
            <queryParams>
              <queryIdentifier>
                <name>Personal Searches/AdvancedSearch6524265</name>
              </queryIdentifier>
              <searchClassIdentifier>ECO</searchClassIdentifier>
              <criteria>[1049] in ('Other')</criteria>
              <visibility>PUBLIC</visibility>
              <searchType>0</searchType>
              <caseSensitive>false</caseSensitive>
            </queryParams>
          </requests>
        </request>
      </createQuery>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <createQueryResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <query>
              <queryIdentifier>
                <name>Personal Searches/AdvancedSearch6504413</name>
                <id>6100910</id>
                <version>1</version>
              </queryIdentifier>
              <searchClassIdentifier>PartsClass</searchClassIdentifier>
              <criteria>[1001] contains 'TE'</criteria>
              <visibility>PRIVATE</visibility>
              <searchType>0</searchType>
              <caseSensitive>false</caseSensitive>
              <resultAttributes>number</resultAttributes>
              <resultAttributes>description</resultAttributes>
              <resultAttributes>lifecyclePhase</resultAttributes>
              <resultAttributes>rev</resultAttributes>
              <resultAttributes>overallCompliance</resultAttributes>
              <resultAttributes>itemType</resultAttributes>
              <owner>admin</owner>
            </query>
          </responses>
        </response>
      </createQueryResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

```
</soapenv:Body>  
</soapenv:Envelope>
```

## loadQuery

**Service** Allows the user to load a query object given a query Identifier. The identifier can be the full Query Path which includes the queryName appended to the parentFolder path.

**Usage** The loadQuery web service operation loads the query given the query id or query name (full name with folder path + name). The response object contains the QueryObjectType objects. Success of the operation may be verified using the status code in the response object..

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadQuery xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <requests>
            <queryIdentifier>
              <name>Personal Searches/AdvancedSearch4940265</name>
            </queryIdentifier>
          </requests>
        </request>
      </loadQuery>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadQueryResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <query>
              <queryIdentifier>
                <name>Personal Searches/AdvancedSearch4940265</name>
                <id>6101016</id>
                <version>1</version>
              </queryIdentifier>
              <searchClassIdentifier>PartsClass</searchClassIdentifier>
              <criteria>[1001] contains 'TE'</criteria>
              <visibility>PRIVATE</visibility>
              <searchType>0</searchType>
              <caseSensitive>>false</caseSensitive>
              <resultAttributes>number</resultAttributes>
              <resultAttributes>description</resultAttributes>
              <resultAttributes>lifecyclePhase</resultAttributes>
              <resultAttributes>rev</resultAttributes>
              <resultAttributes>overallCompliance</resultAttributes>
              <resultAttributes>itemType</resultAttributes>
              <owner>admin</owner>
            </query>
          </responses>
        </response>
      </loadQueryResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```



## executeSavedQuery

**Service** To execute a query that is already created and saved in the Agile system.

**Usage** The executeSavedQuery operation executes a query given the query id or query name (full name with folder path + name). The response object contains the AgileTableType objects. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <executeSavedQuery
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <requests>
            <queryIdentifier>
              <name>Personal Searches/AdvancedSearch8976589</name>
            </queryIdentifier>
            <params>
              <parameter>Electrical</parameter>
            </params>
          </requests>
        </request>
      </executeSavedQuery>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <executeSavedQueryResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <results>
              <tableIdentifier>
                <objectId>6100916</objectId>
                <tableId>-102</tableId>
                <tableName xsi:nil="true"/>
              </tableIdentifier>
            </results>
          </responses>
        </response>
      </executeSavedQueryResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## saveAsQuery

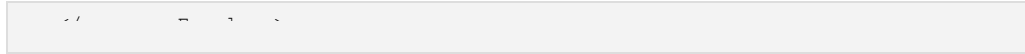
<b>Service</b>	To save an existing query object as a new query object in the Agile system.
<b>Usage</b>	The saveAsQuery operation saves a query given the query id or query name (full name with folder path + name) to new folder or with a new name. The operation will provide mechanism by which the query can be saved into a particular folder. The response will return the new Query Object back to the client. The response object contains the QueryObjectType objects. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <saveAsQuery xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <requests>
            <queryIdentifier>
              <name>Global Searches/QUERY A6120956</name>
            </queryIdentifier>
            <newQueryIdentifier>/Global
Searches/QUERY B3978010</newQueryIdentifier>
            <shortcut>false</shortcut>
          </requests>
        </request>
      </saveAsQuery>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <saveAsQueryResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <query>
              <queryIdentifier>
                <name>/Global Searches/QUERY B3978010</name>
                <id>6101121</id>
                <version>1</version>
              </queryIdentifier>
              <searchClassIdentifier>PartsClass</searchClassIdentifier>
              <criteria>[1001] contains 'TE'</criteria>
              <visibility>PRIVATE</visibility>
              <searchType>0</searchType>
              <caseSensitive>false</caseSensitive>
              <resultAttributes>number</resultAttributes>
              <resultAttributes>description</resultAttributes>
              <resultAttributes>lifecyclePhase</resultAttributes>
              <resultAttributes>rev</resultAttributes>
              <resultAttributes>overallCompliance</resultAttributes>
              <resultAttributes>itemType</resultAttributes>
              <owner>admin</owner>
            </query>
          </responses>
        </response>
      </saveAsQueryResponse>
    </soapenv:Body>

```





## updateQuery

**Service** To update an existing query object in the Agile system.

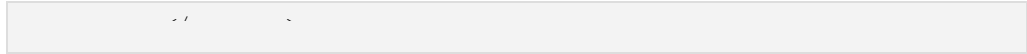
**Usage** The operation updates the query given the query id or query name (full name with folder path + name) with new criteria. The response object contains the QueryObjectType objects. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <updateQuery xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <requests>
            <queryIdentifier>
              <name>Personal Searches/AdvancedSearchNEW1912299</name>
            </queryIdentifier>
            <newQueryParams>
              <queryIdentifier>
                <name>AdvancedSearchNEW1912298</name>
              </queryIdentifier>
              <searchClassIdentifier>ECO</searchClassIdentifier>
              <criteria>[1049] contains ('Other')</criteria>
              <searchType>0</searchType>
              <caseSensitive>>false</caseSensitive>
            </newQueryParams>
          </requests>
        </request>
      </updateQuery>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <updateQueryResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses>
            <query>
              <queryIdentifier>
                <name>/Global Searches/Query A4678815</name>
                <id>6101124</id>
                <version>6</version>
              </queryIdentifier>
              <searchClassIdentifier>PartsClass</searchClassIdentifier>
              <criteria>[1001] contains 'XYZ'</criteria>
              <visibility>PRIVATE</visibility>
              <searchType>0</searchType>
              <caseSensitive>>false</caseSensitive>
              <resultAttributes>number</resultAttributes>
              <resultAttributes>description</resultAttributes>
              <resultAttributes>lifecyclePhase</resultAttributes>
              <resultAttributes>rev</resultAttributes>
              <resultAttributes>overallCompliance</resultAttributes>
              <resultAttributes>itemType</resultAttributes>
              <owner>admin</owner>
            </query>
          </responses>
        </updateQueryResponse>
      </soapenv:Body>
    </soapenv:Envelope>

```



## deleteQuery

<b>Service</b>	To delete an existing query object from the Agile system.
<b>Usage</b>	The deleteQuery operation deletes the query given the query id or query name (full name with folder path + name). Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteQuery xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <request xmlns="">
          <disableAllWarnings>true</disableAllWarnings>
          <requests>
            <queryIdentifier>
              <name>Personal Searches/AdvancedSearch969239</name>
            </queryIdentifier>
          </requests>
        </request>
      </deleteQuery>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteQueryResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Search/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <responses/>
        </response>
      </deleteQueryResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```



## Tables Web Services

This chapter includes the following:

▪ isReadOnlyTable .....	255
▪ clearTable .....	257
▪ copyTable .....	258
▪ addRows .....	259
▪ updateRows .....	260
▪ removeRows .....	261
▪ loadTable .....	262
▪ isFlagSet .....	264

### isReadOnlyTable

**Service** To query a specific Agile Table object and determine if the table status is 'read-only'.

**Usage** The request object consists of class identifier, object id and table identifier that identify the table. The response object returns true or false for read-only status of the table, besides the table name and table display name.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <isReadOnlyTable
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <request xmlns="">
          <isReadOnlyTable>
            <agileTable>
              <classIdentifier>Part</classIdentifier>
              <objectNumber>P00711</objectNumber>
              <tableIdentifier>807</tableIdentifier>
            </agileTable>
          </isReadOnlyTable>
          <isReadOnlyTable>
            <agileTable>
              <classIdentifier>ECO</classIdentifier>
              <objectNumber>C00217</objectNumber>
              <tableIdentifier>809</tableIdentifier>
            </agileTable>
          </isReadOnlyTable>
        </request>
      </isReadOnlyTable>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
```

```
<isReadOnlyTableResponse>
  <response xmlns="">
    <messageId xsi:nil="true"/>
    <messageName xsi:nil="true"/>
    <statusCode>SUCCESS</statusCode>
    <isTableReadOnly>
      <agileTable>
        <classId>10141</classId>
        <className>Part</className>
        <objectId>6112208</objectId>
        <objectName>P00711</objectName>
        <tableId>807</tableId>
        <tableName>Attachments</tableName>
        <tableDisplayName>Attachments</tableDisplayName>
      </agileTable>
      <isReadOnlyTable>false</isReadOnlyTable>
    </isTableReadOnly>
    <isTableReadOnly>
      <agileTable>
        <classId>6141</classId>
        <className>ECO</className>
        <objectId>6112212</objectId>
        <objectName>C00217</objectName>
        <tableId>809</tableId>
        <tableName>AffectedItems</tableName>
        <tableDisplayName>Affected Items</tableDisplayName>
      </agileTable>
      <isReadOnlyTable>false</isReadOnlyTable>
    </isTableReadOnly>
  </response>
</isReadOnlyTableResponse>
</soapenv:Body>
</soapenv:Envelope>
```

## clearTable

**Service** To purge the contents of an Agile Table object by removing all its rows.

**Usage** The request object consists of class identifier and table identifier. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <clearTable xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <request xmlns="">
          <clearTable>
            <agileTable>
              <classIdentifier>Part</classIdentifier>
              <objectNumber>P00707</objectNumber>
              <tableIdentifier>807</tableIdentifier>
            </agileTable>
          </clearTable>
          <clearTable>
            <agileTable>
              <classIdentifier>ECO</classIdentifier>
              <objectNumber>C00216</objectNumber>
              <tableIdentifier>809</tableIdentifier>
            </agileTable>
          </clearTable>
        </request>
      </clearTable>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <clearTableResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </clearTableResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [loadTable](#)

## copyTable

<b>Service</b>	To copy the contents of an Agile Table object from a table to another table.
<b>Usage</b>	The request object consists of relevant information that identifies the tables. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <copyTable xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <request xmlns="">
          <copyTable>
            <sourceTable>
              <classIdentifier>Part</classIdentifier>
              <objectNumber>P00709</objectNumber>
              <tableIdentifier>2000001404</tableIdentifier>
            </sourceTable>
            <targetTable>
              <classIdentifier>Part</classIdentifier>
              <objectNumber>P00710</objectNumber>
              <tableIdentifier>2000001404</tableIdentifier>
            </targetTable>
          </copyTable>
        </request>
      </copyTable>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <copyTableResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </copyTableResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also**      loadTable, clearTable



## addRows

**Service** To add rows in an Agile Table object

**Usage** Uses details of the new row and the table. The request object is built using rowId and objectInfo. Success of the operation is verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addRows xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <request xmlns="">
          <data>
            <objectInfo>
              <classIdentifier>ECO</classIdentifier>
              <objectNumber>C00218</objectNumber>
              <tableIdentifier>809</tableIdentifier>
            </objectInfo>
            <row rowId="0">
              <key1054 attributeId="1054">P00713</key1054>
            </row>
          </data>
        </request>
      </addRows>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <addRowsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </addRowsResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** `getTableMetadata`, `isReadOnlyTable`

## updateRows

**Service** To updated an existing row in an Agile Table object

**Usage** Uses details of the modified row and the table. The request object is built using objectInfo and rowId attributes. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <updateRows xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <request xmlns="">
          <data>
            <objectInfo>
              <classIdentifier>ECO</classIdentifier>
              <objectNumber>C00218</objectNumber>
              <tableIdentifier>809</tableIdentifier>
            </objectInfo>
            <row>
              <rowId>6112255</rowId>
              <row rowId="0">
                <modified element
attributeId="newRev">2</modified element>
              </row>
            </row>
          </data>
        </request>
      </updateRows>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <updateRowsResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </updateRowsResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

**See also** [getTableMetadata](#), [isReadOnlyTable](#), [loadTable](#), [getRowId](#) Method on page 285

## removeRows

<b>Service</b>	To remove an existing row belonging to an Agile Table object
<b>Usage</b>	The request object includes details of the row to be removed and the table identifier. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```

==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeRows xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <request xmlns="">
          <rows>
            <objectInfo>
              <classIdentifier>Part</classIdentifier>
              <objectNumber>P00717</objectNumber>
              <tableIdentifier>803</tableIdentifier>
            </objectInfo>
            <rowId>6112309</rowId>
          </rows>
        </request>
      </removeRows>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <removeRowsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </removeRowsResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

**See also** isReadOnlyTable, loadTable, getRowId Method on page 285

## loadTable

**Service** To load the content of an existing Agile Table object

**Usage** The request object contains identifier of the table to be retrieved and the information to be obtained from it. Success of the operation may be verified using the status code in the response object.

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadTable xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <request xmlns="">
          <tableRequest>
            <classIdentifier>ECO</classIdentifier>
            <objectNumber>C00220</objectNumber>
            <tableIdentifier>809</tableIdentifier>
          </tableRequest>
        </request>
      </loadTable>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <loadTableResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <tableContents>
            <tableIdentifier>
              <classId>6141</classId>
              <className>ECO</className>
              <objectId>6112315</objectId>
              <objectName>C00220</objectName>
              <tableId>809</tableId>
              <tableName>AffectedItems</tableName>
              <tableDisplayName>Affected Items</tableDisplayName>
            </tableIdentifier>
            <row rowId="6112318">
              <objectReferentId>
                <classId>10141</classId>
                <className>Part</className>
                <objectId>6112311</objectId>
                <objectName>P00719</objectName>
              </objectReferentId>
              <hasBeenRedlinedImage xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="6350"
                readOnly="True">false</hasBeenRedlinedImage>
              <itemNumber xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="1054"
                readOnly="False">P00719</itemNumber>
              <attachmentsImage xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="12623"
                readOnly="True">false</attachmentsImage>
            </row>
          </tableContents>
        </response>
      </loadTableResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```



## isFlagSet

**Service** To ascertain the status of a predetermined flag on a particular row of a Agile table.

**Usage** These flags are specific to the table to which the row belongs. For instance, an attachment table may contain a flag 'hasAttachments' denoting whether or not the table has attachments present in it.

To identify the row, object information identifying the table must be coupled with a unique rowId that identifies the row itself. The table may be identified with identifier keys of class, object name and table identifier. The request must also specify the flag pertaining to the current table, for which the row is being queried. Based on this request a boolean value will be obtained in the response with which the nature of the flag status for the current row may be determined..

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <isFlagSet xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
      <request xmlns="">
        <disableAllWarnings>true</disableAllWarnings>
        <requests>
          <objectInfo>
            <classIdentifier>Part</classIdentifier>
            <objectNumber>TEST PARENT7958046</objectNumber>
            <tableIdentifier>-803</tableIdentifier>
            <options>
              <propertyName>redline change</propertyName>
              <propertyValue>TEST ECO8990047</propertyValue>
            </options>
          </objectInfo>
          <rowId>6104164</rowId>
          <flagIdentifier>-103</flagIdentifier>
        </requests>
      </request>
    </isFlagSet>
  </soapenv:Body>
</soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <isFlagSetResponse
xmlns="http://xmlns.oracle.com/AgileObjects/Core/Table/V1">
      <response xmlns="">
        <messageId xsi:nil="true"/>
        <messageName xsi:nil="true"/>
        <statusCode>SUCCESS</statusCode>
        <responses>
          <isFlagSet>true</isFlagSet>
        </responses>
      </response>
    </isFlagSetResponse>
  </soapenv:Body>
</soapenv:Envelope>
```







## User Profile Web Services

This chapter includes the following:

▪ getNotifications .....	267
▪ deleteNotifications .....	276
▪ getMyWorkflowRoutings .....	277

### getNotifications

**Service** Gets all notification for the current User.

**Usage** The operation will retrieve all the notifications pertaining to the current Agile user. The response will return a set of notifications for the user with information describing the subject, the sender and the date when the notification was received.

#### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getNotifications
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/UserProfile/V1">
        <request xmlns=""/>
      </getNotifications>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
HTTP/1.1 200 OK
Date: Wed, 02 Jun 2010 12:08:37 GMT
Server: Oracle Containers for J2EE
Set-Cookie:
JSESSIONID=156e314ab84bdf970e0466b030e455cad5188d6ed8ccc6db6d3647470b646d94;
path=/core
Cache-Control: private
Connection: close
Content-Type: text/xml; charset=utf-8

<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getNotificationsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/UserProfile/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <table>
            <tableIdentifier xsi:nil="true"/>
            <row rowId="6000045">
              <Type xsi:type="common:AgileListEntryType"
                xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
                <selection>
                  <id>1</id>
                  <apiName></apiName>
                </selection>
              </Type>
            </row>
          </table>
        </response>
      </getNotificationsResponse>
    </soapenv:Body>
  </soapenv:Envelope>

```

```

        </selection>
    </Type>
    <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME8203143 with %Allocation: 100.0.</Subject>
    <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Assignment of an Activity
to a Resource</NotificationType>
    <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>18027</classId>
        <className>Program</className>
        <classDisplayName>Program</classDisplayName>
        <objectId>143</objectId>
        <objectName>PGM NAME8203143</objectName>
        <version>2</version>
    </Regarding>
    <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>11610</classId>
        <className>User</className>
        <classDisplayName>User</classDisplayName>
        <objectId>704</objectId>
        <objectName>admin</objectName>
        <version>1</version>
    </SentBy>
    <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
        Activity PGM NAME8203143
        has been assigned to you by Administrator (admin). The scheduled
        dates of the assigned activity are: Scheduled Start: 11/30/2010
        08:00:00 AM Scheduled End: 11/30/2010 05:00:00 PM
        Please access your Agile Inbox to accept or decline assignment to this
        activity: http://MMUTHUSW-
        IDC:8888/web/PLMServlet?action=OpenEmailObject&amp;classid=18022&amp;objid=143
    </Body>
    <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T12:06:36.000Z</Received>
    </row>
    <row rowId="6000044">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <selection>
                <id>0</id>
                <apiName></apiName>
                <value>Notification</value>
            </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME8203143 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Notification for Activity
Assignment to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>18027</classId>
            <className>Program</className>
            <classDisplayName>Program</classDisplayName>
            <objectId>143</objectId>
            <objectName>PGM NAME8203143</objectName>
            <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>11610</classId>
            <className>User</className>
            <classDisplayName>User</classDisplayName>
            <objectId>704</objectId>
            <objectName>admin</objectName>
            <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
            Activity PGM NAME8203143
            has been assigned to you by Administrator (admin). Please access your
            My Assignments to accept or decline assignment to this activity: http://MMUTHUSW-
            IDC:8888/web/PLMServlet?action=OpenEmailObject&amp;classid=18022&amp;objid=143
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T12:06:36.000Z</Received>
    </row>

```

```

    <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
    <selection>
    <id>1</id>
    <apiName></apiName>
    <value>Request</value>
    </selection>
    </Type>
    <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME9699280 with %Allocation: 100.0.</Subject>
    <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Assignment of an Activity
to a Resource</NotificationType>
    <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
    <classId>18027</classId>
    <className>Program</className>
    <classDisplayName>Program</classDisplayName>
    <objectId>136</objectId>
    <objectName>PGM NAME9699280</objectName>
    <version>2</version>
    </Regarding>
    <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
    <classId>11610</classId>
    <className>User</className>
    <classDisplayName>User</classDisplayName>
    <objectId>704</objectId>
    <objectName>admin</objectName>
    <version>1</version>
    </SentBy>
    <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
        Activity PGM NAME9699280
        has been assigned to you by Administrator (admin). The scheduled
        dates of the assigned activity are: Scheduled Start: 11/30/2010
        08:00:00 AM Scheduled End: 11/30/2010 05:00:00 PM
        Please access your Agile Inbox to accept or decline assignment to this
        activity: http://MMUTHUSW-
        IDC:8888/web/PLMServlet?action=OpenEmailObject&amp;classid=18022&amp;objid=136
    </Body>
    <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:40:47.000Z</Received>
    </row>
    <row rowId="6000038">
    <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
    <selection>
    <id>0</id>
    <apiName></apiName>
    <value>Notification</value>
    </selection>
    </Type>
    <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME9699280 with %Allocation: 100.0.</Subject>
    <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Notification for Activity
Assignment to a Resource</NotificationType>
    <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
    <classId>18027</classId>
    <className>Program</className>
    <classDisplayName>Program</classDisplayName>
    <objectId>136</objectId>
    <objectName>PGM NAME9699280</objectName>
    <version>2</version>
    </Regarding>
    <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
    <classId>11610</classId>
    <className>User</className>
    <classDisplayName>User</classDisplayName>
    <objectId>704</objectId>
    <objectName>admin</objectName>
    <version>1</version>
    </SentBy>

```

```

        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activity PGM NAME9699280
has been assigned to you by Administrator (admin). Please access your
My Assignments to accept or decline assignment to this activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=136
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:40:47.000Z</Received>
    </row>
    <row rowId="6000035">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <selection>
                <id>0</id>
                <apiName></apiName>
                <value>Notification</value>
            </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME3078104 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Notification for Activity
Assignment to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>18027</classId>
            <className>Program</className>
            <classDisplayName>Program</classDisplayName>
            <objectId>133</objectId>
            <objectName>PGM NAME3078104</objectName>
            <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>11610</classId>
            <className>User</className>
            <classDisplayName>User</classDisplayName>
            <objectId>704</objectId>
            <objectName>admin</objectName>
            <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
            Activity PGM NAME3078104 has been assigned to you by Administrator
            (admin).
            Please access your My Assignments to accept or decline assignment to this
            activity: http://MMUTHUSW-
            IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=133
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:37:33.000Z</Received>
    </row>
    <row rowId="6000034">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <selection>
                <id>1</id>
                <apiName></apiName>
                <value>Request</value>
            </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME3078104 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Assignment of an Activity
to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>18027</classId>
            <className>Program</className>
            <classDisplayName>Program</classDisplayName>
            <objectId>133</objectId>
            <objectName>PGM NAME3078104</objectName>
            <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>11610</classId>
            <className>User</className>

```

```

</objectName>admin</objectName>
<version>1</version>
</SentBy>
<Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activity PGM NAME3078104
has been assigned to you by Administrator (admin). The scheduled
dates of the assigned activity are: Scheduled Start: 11/30/2010
08:00:00 AM Scheduled End: 11/30/2010 05:00:00 PM
Please access your Agile Inbox to accept or decline assignment to this
activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=133
</Body>
<Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:37:33.000Z</Received>
</row>
<row rowId="6000031">
<Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
<selection>
<id>0</id>
<apiName></apiName>
<value>Notification</value>
</selection>
</Type>
<Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME5181671 with %Allocation: 100.0.</Subject>
<NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Notification for Activity
Assignment to a Resource</NotificationType>
<Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
<classId>18027</classId>
<className>Program</className>
<classDisplayName>Program</classDisplayName>
<objectId>130</objectId>
<objectName>PGM NAME5181671</objectName>
<version>2</version>
</Regarding>
<SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
<classId>11610</classId>
<className>User</className>
<classDisplayName>User</classDisplayName>
<objectId>704</objectId>
<objectName>admin</objectName>
<version>1</version>
</SentBy>
<Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activity PGM NAME5181671
has been assigned to you by Administrator (admin). Please access your
My Assignments to accept or decline assignment to this activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=130
</Body>
<Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:33:59.000Z</Received>
</row>
<row rowId="6000030">
<Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
<selection>
<id>1</id>
<apiName></apiName>
<value>Request</value>
</selection>
</Type>
<Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME5181671 with %Allocation: 100.0.</Subject>
<NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Assignment of an Activity
to a Resource</NotificationType>
<Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
<classId>18027</classId>
<className>Program</className>
<classDisplayName>Program</classDisplayName>
<objectId>130</objectId>
<objectName>PGM_NAME5181671</objectName>

```

```

        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>11610</classId>
        <className>User</className>
        <classDisplayName>User</classDisplayName>
        <objectId>704</objectId>
        <objectName>admin</objectName>
        <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
        Activity PGM NAME5181671
has been assigned to you by Administrator (admin). The scheduled
dates of the assigned activity are: Scheduled Start: 11/30/2010
08:00:00 AM Scheduled End: 11/30/2010 05:00:00 PM
Please access your Agile Inbox to accept or decline assignment to this
activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=130
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:33:59.000Z</Received>
        </row>
        <row rowId="6000029">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <selection>
        <id>1</id>
        <apiName></apiName>
        <value>Request</value>
        </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME9450886 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Assignment of an Activity
to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>18027</classId>
        <className>Program</className>
        <classDisplayName>Program</classDisplayName>
        <objectId>127</objectId>
        <objectName>PGM NAME9450886</objectName>
        <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>11610</classId>
        <className>User</className>
        <classDisplayName>User</classDisplayName>
        <objectId>704</objectId>
        <objectName>admin</objectName>
        <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
        Activity PGM NAME9450886
has been assigned to you by Administrator (admin). The scheduled
dates of the assigned activity are: Scheduled Start: 11/30/2010
08:00:00 AM Scheduled End: 11/30/2010 05:00:00 PM
Please access your Agile Inbox to accept or decline assignment to this
activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=127
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:26:34.000Z</Received>
        </row>
        <row rowId="6000028">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <selection>
        <id>0</id>
        <apiName></apiName>
        <value>Notification</value>
        </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM_NAME9450886 with %Allocation: 100.0.</Subject>

```

```

        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Notification for Activity
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>18027</classId>
        <className>Program</className>
        <classDisplayName>Program</classDisplayName>
        <objectId>127</objectId>
        <objectName>PGM_NAME9450886</objectName>
        <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>11610</classId>
        <className>User</className>
        <classDisplayName>User</classDisplayName>
        <objectId>704</objectId>
        <objectName>admin</objectName>
        <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activity PGM_NAME9450886
has been assigned to you by Administrator (admin). Please access your
My Assignments to accept or decline assignment to this activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&amp;classid=18022&amp;objid=127
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:26:34.000Z</Received>
    </row>
    <row rowId="6000024">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <selection>
        <id>0</id>
        <apiName></apiName>
        <value>Notification</value>
        </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM_NAME2882289 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Notification for Activity
Assignment to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>18027</classId>
        <className>Program</className>
        <classDisplayName>Program</classDisplayName>
        <objectId>124</objectId>
        <objectName>PGM_NAME2882289</objectName>
        <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>11610</classId>
        <className>User</className>
        <classDisplayName>User</classDisplayName>
        <objectId>704</objectId>
        <objectName>admin</objectName>
        <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activity PGM_NAME2882289
has been assigned to you by Administrator (admin). Please access your
My Assignments to accept or decline assignment to this activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&amp;classid=18022&amp;objid=124
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:23:02.000Z</Received>
    </row>
    <row rowId="6000026">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <selection>
        <id>1</id>
        <apiName></apiName>
        <value>Request</value>
        </selection>
        </Type>

```

```

        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME6328082 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Assignment of an Activity
to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>18027</classId>
            <className>Program</className>
            <classDisplayName>Program</classDisplayName>
            <objectId>124</objectId>
            <objectName>PGM NAME2882289</objectName>
            <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>11610</classId>
            <className>User</className>
            <classDisplayName>User</classDisplayName>
            <objectId>704</objectId>
            <objectName>admin</objectName>
            <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
            Activity PGM NAME2882289
            has been assigned to you by Administrator (admin). The scheduled
            dates of the assigned activity are: Scheduled Start: 11/30/2010
            08:00:00 AM Scheduled End: 11/30/2010 05:00:00 PM
            Please access your Agile Inbox to accept or decline assignment to this
            activity: http://MMUTHUSW-
            IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=124
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:23:02.000Z</Received>
    </row>
    <row rowId="6000021">
        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <selection>
                <id>1</id>
                <apiName></apiName>
                <value>Request</value>
            </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME6328082 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Assignment of an Activity
to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>18027</classId>
            <className>Program</className>
            <classDisplayName>Program</classDisplayName>
            <objectId>121</objectId>
            <objectName>PGM NAME6328082</objectName>
            <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
            <classId>11610</classId>
            <className>User</className>
            <classDisplayName>User</classDisplayName>
            <objectId>704</objectId>
            <objectName>admin</objectName>
            <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
            Activity PGM NAME6328082
            has been assigned to you by Administrator (admin). The scheduled
            dates of the assigned activity are: Scheduled Start: 11/30/2010
            08:00:00 AM Scheduled End: 11/30/2010 05:00:00 PM
            Please access your Agile Inbox to accept or decline assignment to this
            activity: http://MMUTHUSW-
            IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=121
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:21:49.000Z</Received>
    </row>

```



```

        <Type xsi:type="common:AgileListEntryType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <selection>
        <id>0</id>
        <apiName></apiName>
        <value>Notification</value>
        </selection>
        </Type>
        <Subject xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Project Activity Assignment:
PGM NAME6328082 with %Allocation: 100.0.</Subject>
        <NotificationType xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">Activities - Notification for Activity
Assignment to a Resource</NotificationType>
        <Regarding xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>18027</classId>
        <className>Program</className>
        <classDisplayName>Program</classDisplayName>
        <objectId>121</objectId>
        <objectName>PGM NAME6328082</objectName>
        <version>2</version>
        </Regarding>
        <SentBy xsi:type="common:AgileObjectType"
xmlns:common="http://xmlns.oracle.com/AgileObjects/Core/Common/V1">
        <classId>11610</classId>
        <className>User</className>
        <classDisplayName>User</classDisplayName>
        <objectId>704</objectId>
        <objectName>admin</objectName>
        <version>1</version>
        </SentBy>
        <Body xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
        Activity PGM NAME6328082
has been assigned to you by Administrator (admin). Please access your
My Assignments to accept or decline assignment to this activity: http://MMUTHUSW-
IDC:8888/web/PLMServlet?action=OpenEmailObject&classid=18022&objid=121
        </Body>
        <Received xsi:type="xs:date"
xmlns:xs="http://www.w3.org/2001/XMLSchema">2010-06-02T11:21:49.000Z</Received>
        </row>
    </table>
</response>
</getNotificationsResponse>
</soapenv:Body>
</soapenv:Envelope>

```

## deleteNotifications

**Service** Deletes a set of notifications for the current User.

**Usage** The operation will delete the specified notifications from a set of notifications populated in the notifications tab for the current user. The request will identify the notification to be removed and the response will ascertain the status of the operation..

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteNotifications
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/UserProfile/V1">
        <request xmlns="">
          <rowId>6000045</rowId>
          <rowId>6000044</rowId>
        </request>
      </deleteNotifications>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <deleteNotificationsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/UserProfile/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
        </response>
      </deleteNotificationsResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

## getMyWorkflowRoutings

**Service** Gets the workflow routings objects for the current User.

**Usage** The operation will retrieve all the workflow routings applicable for the current user as per the set of workflows pending the user's action in the workflow routings tab. The response will return a set of workflow routings for the user with information describing each workflow routing..

### Sample Code SOAP

```
==== Request ====
<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getMyWorkflowRoutings
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/UserProfile/V1">
        <request xmlns="">
          <workflowDateFilter>30</workflowDateFilter>
        </request>
      </getMyWorkflowRoutings>
    </soapenv:Body>
  </soapenv:Envelope>
==== Response ====
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <getMyWorkflowRoutingsResponse
        xmlns="http://xmlns.oracle.com/AgileObjects/Core/UserProfile/V1">
        <response xmlns="">
          <messageId xsi:nil="true"/>
          <messageName xsi:nil="true"/>
          <statusCode>SUCCESS</statusCode>
          <table>
            <tableIdentifier>
              <objectId>309</objectId>
              <tableId>-102</tableId>
              <tableName xsi:nil="true"/>
            </tableIdentifier>
            <row rowId="0">
              <objectReferentId>
                <classId>16437</classId>
                <className>SourcingProject</className>
                <objectId>100001</objectId>
                <objectName xsi:nil="true"/>
              </objectReferentId>
              <Number xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">DELL 101</Number>
              <Description xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3">DELL PROJECT
                101</Description>
              <Status xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="4">Open</Status>
              <Action xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="6">Assigned to
                Me</Action>
              <Workflow xsi:type="xs:string"
                xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="7"></Workflow>
            </row>
            <row rowId="1">
              <objectReferentId>
                <classId>16437</classId>
                <className>SourcingProject</className>
                <objectId>100007</objectId>
                <objectName xsi:nil="true"/>
              </objectReferentId>
```

```

<Number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">DELL 107</Number>
<Description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3">DELL PROJECT
107</Description>
<Status xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="4">Open</Status>
<Action xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="6">Assigned to
Me</Action>
<Workflow xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="7"></Workflow>
</row>
<row rowId="2">
<objectReferentId>
<classId>l6437</classId>
<className>SourcingProject</className>
<objectId>100008</objectId>
<objectName xsi:nil="true"/>
</objectReferentId>
<Number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">DELL 108</Number>
<Description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3">DELL PROJECT
108</Description>
<Status xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="4">Open</Status>
<Action xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="6">Assigned to
Me</Action>
<Workflow xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="7"></Workflow>
</row>
<row rowId="3">
<objectReferentId>
<classId>l6437</classId>
<className>SourcingProject</className>
<objectId>100009</objectId>
<objectName xsi:nil="true"/>
</objectReferentId>
<Number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">DELL 109</Number>
<Description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3">DELL PROJECT
109</Description>
<Status xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="4">Open</Status>
<Action xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="6">Assigned to
Me</Action>
<Workflow xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="7"></Workflow>
</row>
<row rowId="4">
<objectReferentId>
<classId>l6437</classId>
<className>SourcingProject</className>
<objectId>108000</objectId>
<objectName xsi:nil="true"/>
</objectReferentId>
<Number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">PRJSeedData1</Number>
<Description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3">seed
date</Description>
<Status xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="4">Open</Status>
<Action xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="6">Assigned to
Me</Action>
<Workflow xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="7"></Workflow>
</row>
<row rowId="5">
<objectReferentId>
<classId>l6445</classId>
<className>RFQ</className>
<objectId>10001</objectId>
<objectName xsi:nil="true"/>
</objectReferentId>

```

```

        <Number xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="2">RFQ-
RFQ-123456789</Number>
        <Description xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="3">test for
RFQ</Description>
        <Status xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="4">Draft</Status>
        <Action xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="6">Assigned to
Me</Action>
        <Workflow xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema" attributeId="7"></Workflow>
    </row>
</table>
</response>
</getMyWorkFlowRoutingsResponse>
</soapenv:Body>
</soapenv:Envelope>

```



# Appendix

**This Appendix includes the following:**

---

▪ Working with Java Samples.....	281
▪ AddFileSOAPAttachment Method .....	283
▪ Helper Methods .....	285
▪ Troubleshooting .....	291

## Working with Java Samples

The Java Sample Codes covered in this book, and those available for download from eDelivery.oracle.com site, demonstrate various usage characteristics of Agile 93 Web Services. Apart from outlining the basic cases for each Web Service, they also elaborate upon more specific cases that involve usage of options or mandatory message elements.

Download these Sample Codes in 'src' directory. They are categorized into different packages based on the type of Web Service, which are AdminMetaData, Attachment, Business, Collaboration, PC, Search and Table.

Batch files for building and running samples independent of a java IDE are also provided.

## Building Stubs and Compiling the Samples

A batch file **build.bat** has been provided, which is located in the main directory **JavaWeb Servicesamples**. This batch file uses **ant** tasks to generate stubs for Agile Web Services and compiles the java sample files after generation of these stubs.

Along with **build.bat**, you will find a file **custom.properties** file that specifies the basic configuration properties, such as the Agile server URL, username, password of your user and also the URL of DFM. Unless this property file is edited to reflect the values appropriate for your Agile environment, you will not be able to generate stubs or run the samples.

After ensuring that 'custom.properties' has been modified appropriately, run the batch file 'build.bat'. Running the same through a command prompt may help in identifying error statements (if any) that are echoed onto the console. If the message 'BUILD SUCCESSFUL' is displayed on the console then the build process was completed without any errors. However, if the message 'BUILD FAILED' is observed on the console, your 'custom.properties' configuration may be incorrect and you should verify the same.

The stubs and the compiled samples are added to the folder build/built/\*.jar as two jar files 'ws\_samples.jar' and 'ws\_stubs.jar' which are used later while running the sample.

## Executing the Samples using ant Task

After building the stubs and compiling the sample files by following the steps outlined in [Building](#)

[Stubs and Compiling the Samples](#) on page 281 section,

any sample file may be readily executed by using the batch file run.bat and specifying the fully qualified class name as an argument.

Browse the source directory 'JavaWeb Servicesamples/src' to find the package structure of the sample that you are looking for.

For example, to the run the sample 'CreateObjectAPIName' that creates an object using API names, the following command must be executed through command prompt:

```
run business.create.CreateObjectAPIName
```

Similarly, to run a sample from another sample package, say AddRowsSiteSpecific of the table Web Service, use the command:

```
run table.addrows.AddRowsSiteSpecific
```

If no argument is passed to run.bat, the all the available samples will executed sequentially.

---

**Note** While running samples using this ant task, the Agile server URL, username and password properties are retrieved from the same 'custom.properties' file that was used for building stubs.

---

## Executing the Samples using a Java IDE

To run the sample files from a Java IDE, such as JDeveloper or Eclipse, create a new project or workspace in your IDE (as applicable to your IDE) and in your project properties modify / add project source paths to include the 'JavaWeb Servicesamples/src' directory where the sample source code is located.

You will also need to update your project library or classpath information to include all the necessary classes of Axis and other Agile jar files which have been used in the course of sample development.

Ensure that you have added all the jar files under the folders 'JavaWeb Servicesamples/build/axis' and 'JavaWeb Servicesamples/build/lib' to your library / classpath.

Any sample file may now be executed by browsing through the package structure, and running the desired sample.

---

**Note** The static variables relating to Agile server url, username and password in each java sample must be modified if you choose to run the sample through an IDE.

---

## Understanding the Code

Each java sample file contains header documentation at the class level explaining the functionality or usage scenario that the sample demonstrates. A set of static variables relating to server url, username, password and variables specific to that sample, like partNumber or folderName or nextStatus, are declared here.

If the sample is executed using the ant task (via run.bat), then the server configuration related static



variables are overridden by the method `checkArguments(String[] args())`, which obtains arguments from the ant task and reinitializes server URL username, password and DFM URL variables.

In the case of a sample executed through a Java IDE, the server configuration variables must be modified manually to reflect the server settings of your Agile server.

With the exception of `adminMetaData` services, all samples provided here use the method `prepareData()` to prepare all the data prerequisites necessary to create a scenario using which a particular Web Service may be demonstrated meaningfully.

For example, if the sample demonstrates usage of the operation `loadTable` to load a table from a particular version of an Agile object, `prepareData()` will do the following:

1. Create a part object
2. Add a change, modify the part
3. Provide a new version number
4. Release the change.

After the data has been prepared, the operation `loadTable` is now used with the option 'version' to demonstrate the retrieval of a table from a particular version of an Agile object.

If you intend to use your own data or scenario to execute a Web Service sample, comment out the '`prepareData();`' statement in the main method of that sample. After this, edit the static variables at the top of the code and specify your own data.

All the operations performed in data preparation are also achieved using Agile 93 Web Services. To gain a broader understanding of how Web Services are used in conjunction to orchestrate a larger task, examine the file `DataPrepare.java` in the package `src/run/DataPrepare.java`

The sample files are documented with comments at each stage and evince several usage characteristics of these Web Services while illustrating how basic requests are formed and how the responses obtained are used.

## AddFileSOAPAttachment Method

This sample method demonstrates addition of a file attachment to the Attachment Tab of an Agile object using SOAP.

Create the request object `AddFileAttachmentRequestType` for the `addFileAttachment` operation.

Create an array of requests of type `AgileAddFileAttachmentRequestType`. Batch operations may be performed by populating as many request objects as required to add several files to different objects with one single operation.

For each batched request, specify the unique object to whose attachment tab the files are to be added. Supply class identifier and object number information for the same.

The exact specification of the attachment to be added is defined as an object of type `AgileAddFileAttachmentRequestType`. This object includes information about the name of the file and its description and content.

While using SOAP attachments, create a datahandler to specify the file source and add the content as a soap attachment to the soap request. Finally set the `contentId` onto

### AgileAddFileAttachmentRequestType.

The request objects are set and the Agile Stub is used to make the addFileAttachment Web Service call. The status code obtained from the response object is printed to verify the success of the addFileAttachment operation.

If the status code is not 'SUCCESS', then populate the list of exceptions returned by the Web Service.

If the Web Service call was successful, then state the same.

```
try {
    setupServerLogin();

    AddFileAttachmentRequestType addFileAttachmentRequestType =
        new AddFileAttachmentRequestType();
    AgileAddFileAttachmentRequest agileAddFileAttachmentRequest[] =
        new AgileAddFileAttachmentRequest[1];
    agileAddFileAttachmentRequest[0] =
        new AgileAddFileAttachmentRequest();
    agileAddFileAttachmentRequest[0].setClassIdentifier("Part");
    agileAddFileAttachmentRequest[0].setObjectNumber(partNumber);
    System.out.println("Adding a SOAP attachment to the part '" +
        partNumber + "...");
    AgileAddFileAttachmentRequestType attachments[] = new
    AgileAddFileAttachmentRequestType[1];
    attachments[0] =
        new AgileAddFileAttachmentRequestType();
    attachments[0].setName("Filename.txt");
    attachments[0].setDescription("Description for file ");

    String filename = "sample123456.txt";
    BufferedWriter out =
        new BufferedWriter(new FileWriter(filename));
    out.write("Test file");
    out.close();

    DataHandler dh =
        new DataHandler(new FileDataSource(filename));
    AttachmentPart ap = new AttachmentPart(dh);
    agileStub.addAttachment(ap);
    attachments[0].setContentId(ap.getContentId());

    agileAddFileAttachmentRequest[0].setAttachments(attachments);
    agileAddFileAttachmentRequest[0].setSingleFolder(false);

    addFileAttachmentRequestType.setRequests(agileAddFileAttachmentRequest);
    AddFileAttachmentResponseType addFileAttachmentResponseType =
        agileStub.addFileAttachment(addFileAttachmentRequestType);
    System.out.println("\nSTATUS CODE: " +
        addFileAttachmentResponseType.getStatusCode());
    if
    (!addFileAttachmentResponseType.getStatusCode().toString().equals(ResponseStatusCo
    de.SUCCESS.getValue())) {
        AgileExceptionListType[] agileExceptionListType =
            addFileAttachmentResponseType.getExceptions();
        if (agileExceptionListType != null)
            for (int i = 0;
                i < agileExceptionListType.length;
                i++) {
                AgileExceptionType exceptions[] =
                    agileExceptionListType[i].getException();
                for (int j = 0;
                    j < exceptions.length; j++)
                    System.out.println("Exception Id:" +
                        exceptions[j].getExceptionId() +
                        "\nMessage: " +
                        exceptions[j].getMessage());
            }

        AgileWarningListType agileWarningListType[] =
            addFileAttachmentResponseType.getWarnings();
        if (agileWarningListType != null)
            for (int i = 0;
                i < agileWarningListType.length;
                i++) {
```

```

        agileWarningListType[i].getWarning();
        for (int j = 0;
            j < warnings.length; j++)
            System.out.println("Warning Id: " +
                               warnings[j].getWarningId() +
                               "\nMessage: " +
                               warnings[j].getMessage());
    }
} else {
    AgileAddFileAttachmentResponse responses[] =
        addFileAttachmentResponseType.getResponses();
    if (responses != null)
        for (int i = 0; i < responses.length;
            i++) {
            System.out.println("The specified SOAP attachment was successfully
added to the Attachment tab");
            System.out.println("of the object: " +
                               responses[i].getObjectNumber());
        }
}
}
}

```

## Helper Methods

The `getRowId` and `getFileId` are custom helper methods. These are not Agile Web Services operation.

### getRowId Method

**Service**      Obtaining the rowId for a row on an Agile table

**Usage**      Several table and attachment operations require the rowId as input for executing a Web Service. In such cases, the `loadTable` operation is used to load the table that contains the required row and then iterated through the results till the row is found.

To find a particular row in a table, a keyword may be used to search and identify the row. In this example, the filename is used as the key to identify a row.

Search all the rows available in the attachment table and compare all message elements with tag names 'filename' with the filename specified by the client, looking for a match. Once a match is found, the rowId information is derived from the row and returned.

Compare all 'filename' message elements, searching for a match with the filename specified by the user. If a match is found, return either the fileId or rowId based on the requirement.

☞ `getValueFromSelection` is a method written in this sample that handles all message elements of type `AgileListEntryType`. Since 'filename' is a message element of `AgileListEntryType`, the values are elicited by this method. Handle all `AgileListEntryType` message elements, cycle through the selection element, obtain the actual selection value and the selection Id and add it to a `HashMap`. Here the selection value denotes the filename while the selection Id denotes the fileId.

In the case of Filefolders the message element for file name is not an `AgileListEntryType`. The value may be obtained directly.

**Basic Steps** To get a Row ID:

1. Create the request object LoadTableRequestType for the loadTable operation.
2. For each request, specify the table to which the row belongs
3. Tables in Agile Web Services are defined as RequestTableType objects. A specific table may be identified by specifying the class identifier and table identifier attributes.
4. The request objects are set and the agile Stub is used to make the loadTable Web Service call. The status code obtained from the response object is printed to verify the success of the loadTable operation.
5. If the status code is 'SUCCESS', then use the table results to find the required row. Once the row is found, its rowId is returned.
6. Search for the necessary row by using the filename to look for a match and return the rowId.
7. If the status code indicates that the Web Service call was not successful, then populate a list of exceptions.

**Sample Code** getRowId

```

public static int getRowId(String filename, String classIdentifier, String
objectNumber, String tableId){
    try{
        setupServerLogin LoadTable();
        LoadTableRequestType loadTableRequestType = new LoadTableRequestType();

        RequestTableType table[] = new RequestTableType[1];
        table[0] = new RequestTableType();
        table[0].setClassIdentifier(classIdentifier);
        table[0].setObjectNumber(objectNumber);
        table[0].setTableIdentifier( tableId );

        loadTableRequestType.setTableRequest(table);
        LoadTableResponseType loadTableResponseType =
        agileStub Table.loadTable(loadTableRequestType);
        System.out.println("Obtaining row Id / fileId information....");
        System.out.println("STATUS CODE: " + loadTableResponseType.getStatusCode() );

        if( loadTableResponseType.getStatusCode().toString().equals(
ResponseStatusCode.SUCCESS.getValue() ) ){
            AgileTableType[] tables = loadTableResponseType.getTableContents();
            return findRowId(tables, filename);
        }
        else{
            System.out.println("<Failed to load table information>");
            AgileExceptionListType[] agileExceptionListType =
loadTableResponseType.getExceptions();
            if(agileExceptionListType!=null)
                for(int i=0; i<agileExceptionListType.length; i++){
                    AgileExceptionType exceptions[] =
agileExceptionListType[i].getException();
                    for(int j=0; j<exceptions.length; j++){
                        System.out.println(exceptions[j].getMessage() );
                    }

                    AgileWarningListType agileWarningListType[] =
loadTableResponseType.getWarnings();
                    if(agileWarningListType!=null)
                        for( int i=0; i<agileWarningListType.length; i++){
                            AgileWarningType warnings[] = agileWarningListType[i].getWarning();
                            for(int j=0; j<warnings.length; j++){
                                System.out.println("Warning Id: " + warnings[j].getWarningId() +
"\nMessage: " + warnings[j].getMessage() );
                            }
                        }
                    }
        }
    }
}

```

```

        ex.printStackTrace();
    }
    return -1;
}

public static int findRowId(AgileTableType[] tables, String filename){
    if(tables!=null)
        for(int i=0; i<tables.length; i++){
            AgileRowType[] rows = tables[i].getRow();
            if(rows!=null)
                for(int j=0; j<rows.length; j++){
                    MessageElement[] messages = rows[j].get any();
                    for(int m=0; m<messages.length; m++){
                        if(
messages[m].getName().toString().equalsIgnoreCase("filename") ){
                            HashMap fileValues[] =
getValuesFromSelection(messages[m]); for(HashMap fileValue:fileValues)
                                if( fileValue.get("filename").equals(filename) ){
                                    System.out.println("Row Id successfully
retrieved.");
                                    return rows[j].getRowId();
                                }
                            }
                        }
                    }
                }
            }
        }
    return 0;
}

public static HashMap[] getValuesFromSelection(MessageElement element){
    HashMap fileValues[] = null;
    if(element.getType().getLocalPart().equals("AgileListEntryType")){
        AgileListEntryType list = (AgileListEntryType) element.getObjectValue();
        SelectionType selection[] = list.getSelection();
        fileValues = new HashMap [selection.length];
        for(int i=0; i<selection.length; i++){
            fileValues[i] = new HashMap();
            fileValues[i].put("filename", selection[i].getValue() );
            fileValues[i].put("fileid", selection[i].getId() );
        }
    }
    else{
        fileValues = new HashMap [1];
        fileValues[0] = new HashMap();
        fileValues[0].put("filename", element.getFirstChild().getNodeValue() );
        fileValues[0].put("fileId", null );
    }
    return fileValues;
}

```

## getFileId Method

**Service**      Obtaining the FileId for a Row on an Agile Table.

**Usage**          Several attachment operations require the fileId as input for executing. For example, if a particular attachment row has several files associated with it, the fileId is necessary to differentiate between each file available on that row.

In such cases, the we first use the loadTable Web Service to load the table that contains the required row and then iterate through the results till the row has been found. To find a particular row in a table, a keyword may be used to search and identify the row.

In this example, the filename is used as the key to identify a row. Once the filename message element is obtained as an AgileListEntry Type element, the 'id' value of the SelectionType element may be returned. This 'id' tag corresponds to the fileId of that file attachment.

Search all the rows available in the attachment table and compare all message elements with tag names 'filename' with the filename specified by the client, looking for a match. Once a match is found, fileId information is derived from the filename selection elements, whose 'id' corresponds to the fileId.

Compare all 'filename' message elements, searching for a match with the filename specified by the user. If a match is found, return either fileId.

- ☞ If both rowId and fileId values are necessary, then a similar approach may be used to obtain both the rowId and fileId for a particular file attachment from an attachment row.
- ☞ getValueFromSelection is a method written in this sample that handles all message elements of type AgileListEntryType. Since 'filename' is a message element of AgileListEntryType, the values are elicited by this method.

### Usage

Handle all AgileListEntryType message elements, cycle through the selection element, obtain the actual selection value and the selection Id and add it to a HashMap. Here the selection value denotes the filename while the selection Id denotes the fileId.

In the case of Filefolders the message element for file name is not an AgileListEntryType. The value may be obtained directly.

### Basic Steps

To get a File ID:

1. Create the request object LoadTableRequestType for the loadTable operation.
2. For each request, specify the table to which the row belongs. Tables in Agile Web Services are defined as RequestTableType objects. A specific table may be identified by specifying the class identifier and table identifier attributes.
3. The request objects are set and the agile Stub is used to make the loadTable Web Service call. The status code obtained from the response object is printed to verify the success of the loadTable operation.
4. If the status code is 'SUCCESS', then find and retrieve the fileId.
5. Search for the necessary fileId by using the filename to look for a match and return either of the two, as per the requirement specified in the input parameter 'methodType'.
6. If the status code indicates that the Web Service call was not successful, then populate a list of exceptions.

### Sample Code

#### getFileId

```
public static int getFileId(String filename, String clazz, String objectNumber,
String tableId){
    try{
        setupServerLogin LoadTable();
        LoadTableRequestType loadTableRequestType = new LoadTableRequestType();
        RequestTableType table[] = new RequestTableType[1];
        table[0] = new RequestTableType();
        table[0].setClassIdentifier(clazz);
        table[0].setObjectNumber(objectNumber);
        table[0].setTableIdentifier( tableId );
    }
```

```

loadTableRequestType.setTableRequest(table);
LoadTableResponseType loadTableResponseType =
agileStub Table.loadTable(loadTableRequestType);
System.out.println("Obtaining row Id / fileId information.....");
System.out.println("STATUS CODE: " + loadTableResponseType.getStatusCode() );

    if( loadTableResponseType.getStatusCode().toString().equals(
ResponseStatusCode.SUCCESS.getValue() ) ){
        AgileTableType[] tables = loadTableResponseType.getTableContents();
        return findFileId(tables, filename);
    }

    else{
        System.out.println("<Failed to load table information>");
        AgileExceptionListType[] agileExceptionListType =
loadTableResponseType.getExceptions();
        if(agileExceptionListType!=null)
            for(int i=0; i<agileExceptionListType.length; i++){
                AgileExceptionType exceptions[] =
agileExceptionListType[i].getException();
                for(int j=0; j<exceptions.length; j++)
                    System.out.println(exceptions[j].getMessage() );
            }

        AgileWarningListType agileWarningListType[] =
loadTableResponseType.getWarnings();
        if(agileWarningListType!=null)
            for( int i=0; i<agileWarningListType.length; i++){
                AgileWarningType warnings[] = agileWarningListType[i].getWarning();
                for(int j=0; j<warnings.length; j++)
                    System.out.println("Warning Id: " + warnings[j].getWarningId() +
"\nMessage: " + warnings[j].getMessage() );
            }
    }

} catch (Exception ex) {
    ex.printStackTrace();
}
return -1;
}

public static int findFileId(AgileTableType[] tables, String filename){
    if(tables!=null)
        for(int i=0; i<tables.length; i++){
            AgileRowType[] rows = tables[i].getRow();
            if(rows!=null)
                for(int j=0; j<rows.length; j++){
                    MessageElement[] messages = rows[j].get any();
                    for(int m=0; m<messages.length; m++){
                        if(
messages[m].getName().toString().equalsIgnoreCase("filename") ){
                            HashMap fileValues[] =
getValuesFromSelection(messages[m]);
                            for(HashMap fileValue:fileValues)
                                if( fileValue.get("filename").equals(filename) ){
                                    System.out.println("File Id successfully
retrieved.");
                                    return (Integer) fileValue.get("fileid");
                                }
                            }
                    }
                }
        }
    return 0;
}

public static HashMap[] getValuesFromSelection(MessageElement element){
    HashMap fileValues[] = null;
    if(element.getType().getLocalPart().equals("AgileListEntryType")){
        AgileListEntryType list = (AgileListEntryType) element.getObjectValue();
        SelectionType selection[] = list.getSelection();
        fileValues = new HashMap [selection.length];
        for(int i=0; i<selection.length; i++){
            fileValues[i] = new HashMap();
            fileValues[i].put("filename", selection[i].getValue() );
            fileValues[i].put("fileid", selection[i].getId() );
        }
    }
}

```

```
    }
    else{
        fileValues = new HashMap [1];
        fileValues[0] = new HashMap();
        fileValues[0].put("filename", element.getFirstChild().getNodeValue() );
        fileValues[0].put("fileId", null );
    }
    return fileValues;
}
```



## Troubleshooting

**Issue #7639814:** API names are not available for a huge set of lists.

**Resolution:** GetLists Admin service is supported only for Admin Lists whichever can be found in **JavaClient>Admin>Lists**. Some class specific lists are not exposed in Admin Lists.

To get values for these Lists, use GetAttributes Admin service, which will return AttributeType. This, in turn, has availableValues as AdminListType. This list values can be used for updateObject or createObject.

**Issue #8253064:** Group sign-off option cannot be set from Web Services.

**Resolution:** Group sign-off is not supported in the current release.

**Issue #9308516:** *undoRedline* operation error message does not convey the wrong "option name" given.

**Resolution:** If you do not use option "redline\_change", or if it is wrongly entered as "redline\_chg", the correct revision of the item will not be loaded. Only the latest released revision will be loaded. In that revision of the item BOM table, the given row id is invalid and hence the invalid rowId exception is thrown. The behavior seems acceptable.

