



# **Agile Product Lifecycle Management**

Customer Needs Management Web Services  
Guide

v1.2

Part No. E23506-01

December 2011

# Oracle Copyright

Copyright © 1995, 2011, 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 trademarks 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 CNM Web Services .....</b>	<b>1</b>
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
Core Web Services.....	4
Web Services Framework.....	6
Schema Design .....	6
Implementation Approach .....	6
Security .....	7
<b>Getting Started with Agile CNM Web Services .....</b>	<b>9</b>
Prerequisites .....	9
Operating Environment .....	9
Standards Compliance .....	10
Authentication and Performance .....	10
Generating Stubs .....	10
Invoking a Web Service.....	12
Understanding the Web Services Responses.....	12
Response Status Code .....	12
Exceptions and Warnings.....	13
<b>CNM Core Web Services Operations.....</b>	<b>15</b>
createObject.....	17
getObject .....	18
updateObject .....	19
addNotes.....	20
addComments.....	21
addUrlReference.....	22
addAgilePlmReference .....	23
getNotes.....	24
getComments .....	25

getReferences .....	26
getStructure .....	27
getTeam .....	28
<b>Working with Java Samples.....</b>	<b>29</b>
Understanding the Code.....	29
Compiling and Running Samples Using Ant Tasks .....	29
<b>Code Tables.....</b>	<b>31</b>

# 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 CNM Web Services

**This chapter includes the following:**

---

▪ Service-Oriented Architecture (SOA).....	1
▪ About Web Services .....	1
▪ Core Web Services.....	4
▪ Web Services Framework.....	6

This document describes CNM Web services that allow external applications to interact with CNM, and enable data exchange (inbound/outbound) alongside process automation (using engines such as BPEL).

## 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 an object, viewing details of object attributes, or updating the attributes and reference details of an object.

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

The guiding principles of SOA are:

- Self-contained and loosely-coupled
- Well-defined standards-based 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 (SOA)** – 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 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

Agile PLM CNM Web services use industry-standard core technologies. Each core technology listed here is explained in detail in the topics that follow.

- [Web Services Description Language \(WSDL\)](#) on page 2
- [XML and XML Schema](#) on page 3
- [Simple Object Access Protocol \(SOAP\)](#) on page 3

## Web Services Description Language (WSDL)

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

- The WSDL file 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, a WSDL file 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 formatting 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 a 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 XML Schema offers an automated mechanism for validating the XML documents. 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:

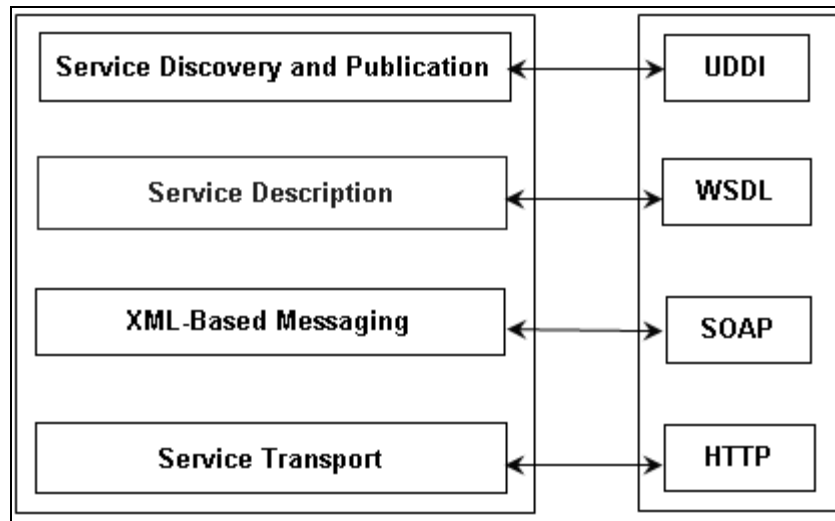
- An envelope for wrapping messages, including addressing and security information
- A set of serialized rules for encoding data types in XML
- Conventions for a procedure call and/or response

## 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 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.



## Core Web Services

The following table lists the Application (Core) Web services for Business Objects in CNM 1.2 that enable you to create, retrieve and update CNM objects.

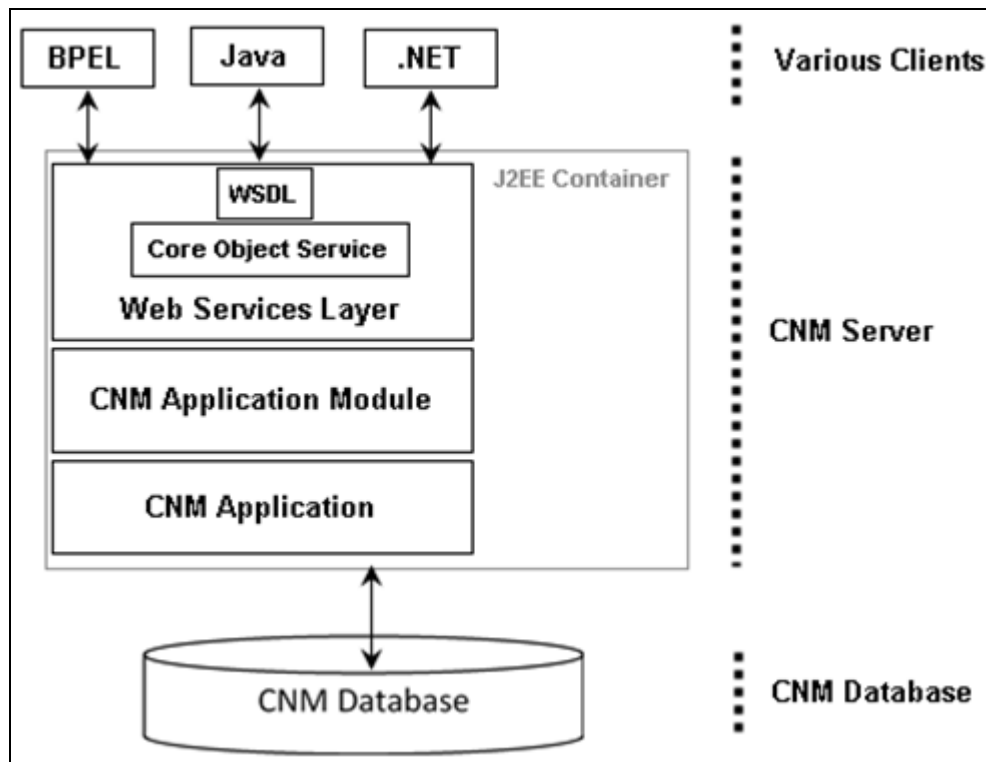
**Note** Objects in Agile CNM consist of Ideas, Quotes, Requirements, or any other object name your administrator configures in the system. For more information, refer to *Oracle Agile PLM Customer Needs Management User and Administration Guide*.

Operations	Web Services (API name)	Purpose
Object Create/Read/Update	Create Object (createObject)	Create a specific CNM object in the CNM system.
	Modify Object Attributes (updateObject)	Update a specific CNM object in the CNM system.
	Read Object Attributes (getObject)	Retrieve a specific CNM object from the CNM system.

Create/Read for related information such as Notes, Comments, and References without Attachments	Read Notes (getNotes)	Retrieve existing notes of a CNM object.
	Read Comments (getComments)	Retrieve existing comments of a note in the CNM object.
	Read References (getReferences)	Retrieve existing references in the CNM object.
	Add Notes (addNotes)	Add new notes to an existing CNM object using relevant information, including details of the new note.
	Add Comments (addComments)	Add new comments to an existing CNM object using relevant information, including details of the new comment.
	Add URL References (addUrlReferences)	Add new URL references to an existing CNM object using relevant information, including details of the new URL reference.
	Add Agile PLM References (addAgilePlmReferences)	Add new Agile PLM references to a CNM object using relevant information, including details of the new Agile PLM reference.
Read for related information such as Structure and Team	Read Structure (getStructure)	Retrieve existing structure details of a CNM object.
	Read Team (getTeam)	Retrieve existing team details of a CNM object.

## Web Services Framework

The CNM Core Web services are bundled with the CNM application. The Web service infrastructure is built on top of existing application modules to expose operations in the CNM application as Web services.



## Schema Design

A generic schema design defines the CNM data model, in line with the CNM representation of data.

Advantages of the generic schema design approach are:

- It provides more flexibility in the server to define the schema at a very high level.
- It allows similar modeling of both out-of-the-box metadata and user-defined metadata.

## Implementation Approach

The top-down approach is used to implement Web services in CNM 1.2. Highlights of this approach are:

1. The interface (WSDL) and schema (XSD) required for Web services are defined before implementing the Web services.

This approach forces attention on:

- The service (design and granularity) instead of implementation.
  - A well-defined standards-based discoverable interface
  - Modularized Schema (XSD) and WSDL for easy maintenance
  - Standards-based WSDL to ensure compatibility across various clients (.NET, Java, and BPEL)
2. Client-side stubs are generated from the WSDL using JAX-WS proxy to implement the functionality.
    - Direct manipulation of the public WSDL file ensures easier backwards compatibility and cross-client compatibility.
    - The XML-based Web service framework ensures compatibility across client technologies.
    - Web service versioning is enabled for backward compatibility.
  3. The generated stubs delegate all calls to a new layer that has the actual implementation of the Web services.
    - The separation between the Web services infrastructure layer and the implementation layer ensures that most porting changes are localized, thus reducing the cost of porting the service to other infrastructures in the future.
    - Users do not have to change their existing code - changes made are backwards compatible and allow easy migration.

## Security

CNM Web service operations follow the same security restrictions as the CNM application. For example, if a user with read-only privilege tries to modify an object's attributes, the Web service returns an error. You also require valid user credentials to access the application server. For more information, refer to *Oracle Agile PLM Customer Needs Management User and Administration Guide*.

CNM Web services support Web Services Security (WS-Security) 1.1.

The following sample demonstrates the generated stubs being initialized for a Business Object Web services client.

```
public static BusinessObject_PortType getBusinessObject(BusinessObject
businessObject, String URL, String username, String password) {
    try{
        businessObject =
            new BusinessObject(new URL(URL), new
QName ("http://xmlns.oracle.com/CNMServices/Core/Business/V1",
"BusinessObject")); }catch(Exception e){};
        SecurityPoliciesFeature securityFeatures =
            new SecurityPoliciesFeature(new String[] {
"oracle/wss_username_token_client_policy" });
        BusinessObject_PortType businessObject_PortType =
businessObject.getBusinessObject(securityFeatures);
```

```
    Map<String, Object> reqContext =  
    ((BindingProvider)businessObject_PortType).getRequestContext();  
    reqContext.put(BindingProvider.USERNAME_PROPERTY, username);  
    reqContext.put(BindingProvider.PASSWORD_PROPERTY, password);  
    return businessObject_PortType;  
}
```

# Getting Started with Agile CNM Web Services

**This chapter includes the following:**

▪ Prerequisites .....	9
▪ Authentication and Performance .....	10
▪ Generating Stubs .....	10
▪ Invoking a Web Service .....	12
▪ Understanding the Web Services Responses .....	12
▪ Exceptions and Warnings .....	13

**To generate Web services from a WSDL file:**

1. Set up your operating environment.  
Refer to the section [Prerequisites](#) on page 9.
2. Obtain the WSDL URL and user credentials from your CNM authentication provider.
3. Authenticate your Web services session.  
Refer to the section [Authentication and Performance](#) on page 10.
4. Build stubs from the WSDL file using the JAX-WS proxy generation method.  
Refer to the section [Generating Stubs](#) on page 10.
5. Run the Web service using a suitable client.  
Refer to the section [Invoking a Web Service](#) on page 12.

## Prerequisites

### Operating Environment

Dependent Application	Version
Agile PLM CNM Server	1.2
Oracle JDeveloper	11.1.1.5.0
Java Development Kit (JDK)	1.6
Java API for XML-based Web services (JAX-WS)	2.0
WebLogic Application Server 11gR1	10.3.5

For information on installation procedures, refer to the *Agile PLM Customer Needs Management Implementation Guide*.

## Standards Compliance

CNM 1.2 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>
SOAP Message Transmission Optimization Mechanism (MTOM)	<a href="http://www.w3.org/TR/soap12-mtom/">http://www.w3.org/TR/soap12-mtom/</a>
JAX-WS 2.0/2.1/2.2 (JSR 224)	<a href="http://www.oracle.com/us/technologies/java/jax-ws-2-141894.html">http://www.oracle.com/us/technologies/java/jax-ws-2-141894.html</a>
WS-Security	<a href="http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss">http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss</a>

## Authentication and Performance

CNM Web services components can be broadly classified into:

- Client-side library (**BusinessObject-Client.jar**)
- Server-side implementation class (**BusinessObject\_PortTypeImpl**)

To connect to the application server, generate the client-side library, and use CNM Web services, you require credentials assigned by the CNM authentication provider. For information on obtaining credentials, refer to *Oracle Agile PLM Customer Needs Management User and Administration Guide*.

To call a Web service, you must generate a Web service proxy (a local object that exposes the same set of methods as the target Web service), and set the credentials and authentication details on the proxy. These credentials are then passed with each Web service call using SOAP Headers. The process of calling Web services is described later in this chapter.

For Web service applications to access data in XML format directly from a Java application, the XML content needs to be converted to a format that is readable by the Java application. JAX-WS uses Java Architecture for XML Binding (JAXB) to manage all data binding tasks. For more information on data binding, refer to the WebLogic documentation on OTN.

## Generating Stubs

A 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 adds network logic to ensure a reliable communication channel between client and server.

**To generate client stubs using the JAX-WS proxy generation method:**

1. Download the CNM 1.2 Web services Java samples to a local folder.
2. Navigate to the directory **JavaSamples\WebServiceProxy**.
3. Locate the batch file **genproxy.bat** and the Ant configuration file **build.properties** in the **WebServiceProxy** folder.
4. Edit the files to ensure they reflect the values appropriate for your CNM environment, specifically, the locations of JDK 1.6 home directory and the WSDL document.

---

**Note** If the **build.properties** and **genproxy.bat** files do not reflect the values appropriate for your CNM environment, you will not be able to generate stubs.

---

5. Run **genproxy.bat**.

Running the batch file through a command prompt console may help you identify error statements (if any).

If the message '**BUILD SUCCESSFUL**' is displayed on the console, it implies the build process was completed without any errors. On the contrary, if the message '**BUILD FAILED**' is displayed on the console, the configuration details in the **build.properties** file may be incorrect and you should verify the same.

The following sample code demonstrates a successful build process.

```
<Local directory>\JavaSamples\WebServiceProxy>genproxy.bat
<Local directory>\JavaSamples\WebServiceProxy>echo on
<Local directory>\JavaSamples\WebServiceProxy>set ANT_HOME=..\lib
<Local directory>\JavaSamples\WebServiceProxy>set JAVA_HOME=<Local
drive>\Oracle\Middleware\jdk160_24
<Local directory>\JavaSamples\WebServiceProxy>set ANT_ARGS=<Local
directory>\JavaSamples\WebServiceProxy>D:\Oracle\Middleware\jdk160_24\b
in\java -cp..\lib\ant.jar;..\lib\ant-launcher.jar;..\lib\ant-nodeps.jar
org.apache.tools.ant.Main
Buildfile: build.xml
clean:
    [delete] Deleting directory
<Local directory>\JavaSamples\WebServiceProxy\src
    [delete] Deleting directory
<Local directory>\JavaSamples\WebServiceProxy\classes
init:
    [mkdir] Created dir:
<Local directory>\JavaSamples\WebServiceProxy\src
    [mkdir] Created dir:
<Local directory>\JavaSamples\WebServiceProxy\classes
generate-proxy:
    [exec] parsing WSDL...
```

```
[exec] generating code...
[exec] compiling code...
generate-proxy-jar:
[jar] Building jar:
<Local directory>\JavaSamples\lib\BusinessObject-Client.jar
all:
BUILD SUCCESSFUL
```

The batch file uses Ant tasks to generate stubs for CNM Web services and creates the JAR file **BusinessObject-Client.jar** under **JavaSamples\lib**. The JAR file is used while running the Java samples.

The xml files **BusinessObject.wsdl** and **BusinessObjectSchema.xsd** containing API documentation are created in the directory **JavaSamples\wsdl**.

## Invoking a Web Service

Invoking a Web service refers to the actions that a client application performs to use the Web service. Client applications that invoke Web services can be written using any technology: Java, Microsoft .NET, and so on.

To invoke CNM Web services in the Java samples provided, ensure that your Java code contains valid user credentials and the correct WSDL location. For more information, refer to the Appendix [Working with Java Samples](#) on page 29.

In addition to the command-line tools, you can use an IDE, such as Oracle JDeveloper, to invoke Web services.

## Understanding the Web Services Responses

### Response Status Code

The response obtained from every Web service call contains a **ResponseStatusCode**, 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 all operations worked as intended.
- **FAILURE** - indicates that all Web services in the batch failed during execution and the intended operations were not performed.
- **WARNING** - indicates that while Web services in the batch were successfully executed, certain warnings were also encountered during the execution. These warnings need to be analyzed to verify if 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 **ResponseStatusCode** 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 manually check whether the operation was successful or not.

The exceptions require code correction or system administration.

The response header for Web services calls consists of a list of exceptions and warnings populated as **CnmExceptionListType** and **CnmWarningListType** 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 element requested in the batch that was the source of the exception(s) or warning(s).

For batch requests, the request index number is used to group the exceptions (for example, if there are two requests and the first one throws exceptions, then the id is equal to "0").

For information on response objects of a particular Web service, refer to the chapter [CNM Core Web Services Operations](#) on page 15.



# CNM Core Web Services Operations

This chapter includes the following:

---

▪ createObject .....	16
▪ getObject .....	17
▪ updateObject .....	19
▪ addNotes .....	20
▪ addComments .....	20
▪ addUrlReference.....	21
▪ addAgilePlmReference .....	22
▪ getNotes .....	24
▪ getComments .....	24
▪ getReferences .....	25
▪ getStructure .....	26
▪ getTeam.....	27

This chapter describes the elements of Agile CNM Business Objects Web services, and provides sample code snippets.

For related information on working with CNM Objects, refer to *Agile PLM Customer Needs Management User and Administration Guide*.

In the Agile CNM Web services framework, all operations require one Request object as input and return one Response object.

- The request contains attribute values used to search or create a CNM object.
- The response contains the data of the respective objects.

Each operation has its own request and response data types, which are inherited from **CnmRequestHeader** and **CnmResponseHeader** respectively.

---

**Note** The request header for the Web service **createObject** is not inherited from **CnmRequestHeader** as the object number is generated by CNM at the time of object creation.

---

**CnmRequestHeader** has the following elements:

<b>classId</b>	<p>The class identifier of the CNM object. This is a required field.</p> <p>Class Identifier codes include:</p> <ul style="list-style-type: none"><li>▫ <b>ClassIdentifierCode.IDEA</b></li><li>▫ <b>ClassIdentifierCode.QUOTE</b></li><li>▫ <b>ClassIdentifierCode.REQUIREMENT</b></li></ul>
<b>objectNumber</b>	<p>The unique number assigned to the specific CNM object type. This is a required field.</p> <p><b>Note</b> Two different type of objects (for example: Idea and Requirement) could have same object number.</p>
<b>inputParameters</b>	<p>Property name and value pair for addressing future requirements.</p> <p>This is an optional field.</p>

**CnmResponseHeader** has the following elements:

<b>responseStatusCode</b>	<p>The results of executing the Web service operation, denoting either successful execution or any of the failed states.</p> <p>Response Status Codes include:</p> <ul style="list-style-type: none"><li>▫ <b>ResponseStatusCode.SUCCESS</b></li><li>▫ <b>ResponseStatusCode.FAILURE</b></li><li>▫ <b>ResponseStatusCode.PARTIAL_SUCCESS</b></li><li>▫ <b>ResponseStatusCode.WARNING</b></li></ul>
<b>exceptions</b>	<p>List of exceptions associated with the execution of a Web service operation.</p>
<b>warnings</b>	<p>List of warnings associated with the execution of a Web service operation.</p>
<b>outputParameters</b>	<p>Optional property name and value pair for addressing future requirements.</p>

# createObject

**Service** To create a specific CNM object in the CNM system.

**Usage** The object specifications are detailed in the request object. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmCreateObjectRequest

Request object for the createObject operation. It contains information relating to the class type and object "name" for creating a CNM object, and additional attributes which might be mandatory, specific to the type of object being created. The object number which uniquely identifies a CNM object of specific type is automatically generated by CNM and returned in the createObject request.

@classId	The class identifier of the CNM object. This is a required field.
@objectName	The name of the object. This is a required field.
@description	The description of the object.
@inputParameters	Optional name and value pair for addressing future requirements

**Response Type** CnmCreateObjectResponse

Response object for the 'createObject' operation. It contains class identifier and object number which uniquely identify the object created by the Web service.

It extends CnmResponseHeader.

@classId	The class identifier of the CNM object
@objectNumber	Unique object number assigned by the system

## Sample Code

```
CnmCreateObjectRequest req = factory.createCnmCreateObjectRequest();
req.setClassId(classCode);
req.setObjectName(objectName);
req.setDescription(description);
CnmCreateObjectResponse resp = businessObjectPortType.createObject(req);
```

## getObject

**Service** To retrieve a specific CNM object in the CNM system.

**Usage** The specifications of the object to be retrieved are detailed in the request object. Comprehensive information about the object is retrieved in the response object after successful execution of the Web service call.

**Request Type** CnmGetObjectRequest

Request object for the 'getObject' operation. It extends CnmRequestHeader.

**Response Type** CnmGetObjectResponse

Response object for the 'getObject' operation. It contains the visible summary details, attributes and values of a CNM object retrieved by the Web service. It extends CnmResponseHeader.

@classId	Class identifier of the CNM object
@objectNumber	Unique number assigned to the CNM object
@objectName	Name of the object
@description	Description of the object
@status	Status of the object
@tags	Tags associated with the object
@dateCreated	Date of creation for the object
@dateModified	Date of modification for this object
@createdBy	User who created the object
@modifiedBy	User who modified the object
@customer	Customers associated with the object
@productLine	Product lines associated with the object
@text01-text15	15 flex text fields
@list01-list15	15 flex list fields
@number01-number15	15 flex number fields
@date01-date15	15 flex date fields

### Sample Code

```
CnmGetObjectRequest req = factory.createCnmGetObjectRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
CnmGetObjectResponse resp = businessObjectPortType.getObject(req);
```



## updateObject

**Service** To update a specific CNM object in the CNM system.

**Usage** The revised object specifications are detailed in the request object where data specific to an CNM object may be expressed. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmUpdateObjectRequest

Request object for the 'updateObject' operation. It contains information relating to the class type and object number for updating a unique CNM Object and additional attributes specifying the fields which are to be updated. It extends CnmRequestHeader.

@objectName	Name of the object
@description	Description of the object
@status	Status of the object
@tags	Tags associated with the object
@customer	Customers associated with the object
@productLine	Product lines associated with the object
@text01-text15	15 flex text fields
@list01-list15	15 flex list fields
@number01-number15	15 flex number fields
@date01-date15	15 flex date fields

**Response Type** CnmUpdateObjectResponse

Response object for the 'updateObject' operation. It extends CnmResponseHeader.

### Sample Code

```
CnmUpdateObjectRequest req = factory.createCnmUpdateObjectRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
req.setObjectName(factory.createCnmUpdateObjectRequestObjectName((String)
objectName));
req.setStatus(factory.createCnmUpdateObjectRequestStatus((String)status)
);
req.setDescription(factory.createCnmUpdateObjectRequestDescription((String)
description));
req.setTags(factory.createCnmUpdateObjectRequestTags((String)tags));
req.setCustomer(factory.createCnmUpdateObjectRequestCustomer((String)cus
tomer));
req.setProductLine(factory.createCnmUpdateObjectRequestProductLine((String)
productLine));
BigDecimal number = BigDecimal.valueOf(1234.56);
req.setNumber01(factory.createCnmUpdateObjectRequestNumber01(number));
```

```
XMLGregorianCalendar xmlDate = new XMLGregorianCalendarImpl();

xmlDate.setYear(2011);
xmlDate.setMonth(10);
xmlDate.setDay(12);
xmlDate.setHour(17);
xmlDate.setMinute(30);
xmlDate.setSecond(25);

req.setDate01(factory.createCnmUpdateObjectRequestDate01(xmlDate));
CnmUpdateObjectResponse resp =
businessObject_PortType.updateObject(req);
```

## addNotes

**Service** To add new notes to an existing CNM object using relevant information that includes details of the new note.

**Usage** The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmAddNotesRequest

Request object for the 'addNotes' operation. It extends CnmRequestHeader.

@noteText	Text details about the note. This is a required field.
-----------	--------------------------------------------------------

**Response Type** CnmAddNotesResponse

Response object for the 'addNotes' operation. This response provides a list of identifiers for the notes that were added by the Web service. It extends CnmResponseHeader.

@noteIds	A list of note identifiers
----------	----------------------------

### Sample Code

```
CnmAddNotesRequest req = factory.createCnmAddNotesRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
List<String> notes = req.getNoteText();
List<String> noteText = new ArrayList<String>();
noteText.add("Add note from Web service.");
notes.addAll(noteText);
CnmAddNotesResponse resp = businessObjectPortType.addNotes(req);
```

## addComments

<b>Service</b>	To add new comments to an existing CNM object using relevant information that includes details of the new comment.	
<b>Usage</b>	The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.	
<b>Request Type</b>	<u>CnmAddCommentsRequest</u>	
	Request object for each of the batch requests for the 'addComments' operation. It extends CnmRequestHeader.	
	@noteId	A unique attribute by which notes are differentiated. This is a required field.
	@commentText	Text details about the comment. This is a required field.
<b>Response Type</b>	<u>CnmAddCommentsResponse</u>	
	Response object for the 'addComments' operation. This response provides a list of identifiers for the comments that were added by the Web service. It extends CnmResponseHeader.	
	@commentIds	A list of comment identifiers

### Sample Code

```
CnmAddCommentsRequest req = factory.createCnmAddCommentsRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
req.setNoteId(noteId);
List<String> comments = req.getCommentText();
List<String> commentText = new ArrayList<String>();
commentText.add("Add comment from Web service.");
comments.addAll(commentText);
CnmAddCommentsResponse resp = businessObjectPortType.addComments(req);
```

## addUrlReference

**Service** To add new URL references to an existing CNM object using relevant information that includes details of the new URL reference.

**Usage** The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmAddUrlReferencesRequest

Request object for each of the batch requests for the 'addUrlReferences' operation. It extends CnmRequestHeader.

@urlReferences	Details about the URL references, including:
@url	The actual URL
@description	URL description [optional]

**Response Type** CnmAddUrlReferencesResponse

Response object for the 'addUrlReferences' operation. This response provides a list of identifiers for the URL references that were added by the Web service. It extends CnmResponseHeader.

@referenceIds	A list of reference identifiers.
@referenceTypeId	The type identifier of the reference. (ReferenceTypeIdIdentifierCode can include URL, AgilePlm, or File)

### Sample Code

```
CnmAddUrlReferencesRequest req =
factory.createCnmAddUrlReferencesRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
List<CnmAddUrlReferenceType> refs = req.getUrlReferences();
List<CnmAddUrlReferenceType> url = new
ArrayList<CnmAddUrlReferenceType>();
CnmAddUrlReferenceType cnn = factory.createCnmAddUrlReferenceType();
cnn.setUrl("http://www.cnn.com");
url.add(cnn);
refs.addAll(url);
CnmAddUrlReferencesResponse resp =
businessObjectPortType.addUrlReferences(req);
```

## addAgilePlmReference

**Service** To add new Agile PLM references to a CNM object using relevant information that includes details of the new Agile PLM reference.

**Usage** The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmAddAgilePlmReferencesRequest

Request object for each of the batch requests for the 'addAgilePlmReferences' operation. It extends CnmRequestHeader.

@agilePlmReferences	Details about the Agile PLM object references, including:										
@baseClass	API name of the Agile class. The different types (class) of Agile PLM references and their API names are tabulated below.										
	<table> <tr> <th>AgilePLM Reference Type</th><th>baseClass (API name)</th></tr> <tr> <td>Items</td><td>ItemsBaseClass</td></tr> <tr> <td>Projects</td><td>ProjectsBaseClass</td></tr> <tr> <td>Problem Reports</td><td>ProblemReportsClass</td></tr> <tr> <td>Prices</td><td>PricesBaseClass</td></tr> </table>	AgilePLM Reference Type	baseClass (API name)	Items	ItemsBaseClass	Projects	ProjectsBaseClass	Problem Reports	ProblemReportsClass	Prices	PricesBaseClass
AgilePLM Reference Type	baseClass (API name)										
Items	ItemsBaseClass										
Projects	ProjectsBaseClass										
Problem Reports	ProblemReportsClass										
Prices	PricesBaseClass										
@objectNumber	Object number in Agile										

**Response Type** CnmAddAgilePlmReferencesResponse

Response object for the 'addAgilePlmReferences' operation. This response provides a list of identifiers for the Agile PLM object references that were added by the Web service. It extends CnmResponseHeader.

@referenceIds	A list of reference identifiers
@referenceTypeId	The type identifier of the reference
	ReferenceTypeIdIdentifierCode can include URL, AgilePlm, or File

### Sample Code

```
CnmAddAgilePlmReferencesRequest req =
factory.createCnmAddAgilePlmReferencesRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
List<CnmAddAgilePlmReferenceType> refs = req.getAgilePlmReferences();
List<CnmAddAgilePlmReferenceType> agileObj = new
ArrayList<CnmAddAgilePlmReferenceType>();
CnmAddAgilePlmReferenceType item =
factory.createCnmAddAgilePlmReferenceType();
item.setBaseClass(itemsBaseClass);
item.setObjectNumber(itemNumber1);
```

```
agileObj.add(item);  
refs.addAll(agileObj);  
CnmAddAgilePlmReferencesResponse resp =  
businessObjectPortType.addAgilePlmReferences(req);
```

## getNotes

**Service** To load the content of existing notes of a CNM object.

**Usage** The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmGetNotesRequest

Request object for the 'getNotes' operation. It extends CnmRequestHeader.

**Response Type** CnmGetNotesResponse

Response object for the 'getNotes' operation. This response provides information about the notes that were loaded by the Web service. It extends CnmResponseHeader.

@notes	A list of notes including following information:
@noteId	Note identifier
@noteText	Text details of the note
@createdBy	User who created the note
@dateCreated	Date of creation of the note

### Sample Code

```
CnmGetNotesRequest req = factory.createCnmGetNotesRequest();  
req.setClassId(classCode);  
req.setObjectNumber(objectNumber);  
CnmGetNotesResponse resp = businessObjectPortType.getNotes(req);
```

## getComments

<b>Service</b>	To load the content of existing comments of a note in the CNM object.	
<b>Usage</b>	The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.	
<b>Request Type</b>	<u>CnmGetCommentsRequest</u>	
	Request object for the 'getComments' operation. It extends CnmRequestHeader.	
	@noteId	An integer and unique attribute by which notes are differentiated.
<b>Response Type</b>	<u>CnmGetCommentsResponse</u>	
	Response object for the 'getComments' operation. This response provides information about the comments that were loaded by the Web service. It extends CnmResponseHeader.	
	@comments	A list of comments including following information:
	@commentId	Comment identifier
	@commentText	Text details of the comment
	@dateCreated	Date of creation of the comment
	@createdBy	User who created the comment

### Sample Code

```
CnmGetCommentsRequest req = factory.createCnmGetCommentsRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
req.setNoteId(noteId);
CnmGetCommentsResponse resp = businessObjectPortType.getComments(req);
```

## getReferences

**Service** To load the content of existing references in the CNM object.

**Usage** The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmGetReferencesRequest

Request object for the 'getReferences' operation. It extends CnmRequestHeader.

**Response Type** CnmGetReferencesResponse

Response object for the 'getReferences' operation. This response provides information about the references that were loaded by the Web service. It extends CnmResponseHeader.

@references	A list of references including:
@referenceTypeId	Reference type identifier
@referenceId	Reference identifier
@name	URL or filename or object number
@baseClass	For Agile PLM reference only: API name of the Agile class
@size	For File reference only: file size
@description	Description of reference
@dateModified	Date of (reference) modification
@modifiedBy	User who modified the object
@fulfillment	Fulfillment field

Null is expected when optional fields are not set.

### Sample Code

```
CnmGetReferencesRequest req = factory.createCnmGetReferencesRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
CnmGetReferencesResponse resp =
businessObjectPortType.getReferences(req);
```



## getStructure

<b>Service</b>	To load the content of an existing structure in the CNM object.
<b>Usage</b>	The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.
<b>Request Type</b>	<u>CnmGetStructureRequest</u> Request object for the 'getStructure' operation. It extends CnmRequestHeader.
<b>Response Type</b>	<u>CnmGetStructureResponse</u> Response object for the 'getStructure' operation. This response provides information about the structure that was loaded by the Web service. It extends CnmResponseHeader.

@objects	A list of objects in the structure including:
@classId	Object type of parent
@objectNumber	Unique object number of a parent assigned by the system
@objectName	Name of the object
@status	Status of the object

The design supports single-level structure in the response.

```
CnmGetStructureRequest req = factory.createCnmGetStructureRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
CnmGetStructureResponse resp = businessObjectPortType.getStructure(req);
```

## getTeam

**Service** To load the content of existing team in the CNM object.

**Usage** The request object is built using information about the CNM object. Success of the operation may be verified using the status code in the response object.

**Request Type** CnmGetTeamRequest

Request object for the 'getTeam' operation. It extends CnmRequestHeader.

**Response Type** CnmGetTeamResponse

Response object for the 'getTeam' operation. This response provides information about the team members that were loaded by the Web service. It extends CnmResponseHeader.

@members	A list of members in the team including:
@teamMemberTypeId	Team member type identifier. TeamMemberIdentifierTypeCode can include User and Group.
@teamMemberId	Team member identifier
@memberName	Name of member
@currentReviewStatus	Current review status
@previousReviewStatus	Previous review status
@requestedBy	User who requested the review
@requestDate	Date of review request
@signoffDate	Date of signoff

Null is expected when optional fields are not set.

### Sample Code

```
CnmGetTeamRequest req = factory.createCnmGetTeamRequest();
req.setClassId(classCode);
req.setObjectNumber(objectNumber);
CnmGetTeamResponse resp = businessObjectPortType.getTeam(req);
```

# Working with Java Samples

**This Appendix includes the following:**

---

- Understanding the Code ..... 29
- Compiling and Running Samples Using Ant Tasks ..... 29

The Java samples in this section demonstrate the various uses of CNM 1.2 Web services. You can download the source code from the [OTN Sample Code Index](http://www.oracle.com/technetwork/indexes/samplecode/agileplm-sample-520945.html) <http://www.oracle.com/technetwork/indexes/samplecode/agileplm-sample-520945.html>.

There are four Use Case samples placed in individual directories. The sub-directory 'src' in each of these directories contains the respective source code. You can build and run these samples using the batch files and Ant build files provided.

---

**Note** You need to use the CNM web application to prepare data for some of the samples. The header documentation lists the prerequisites to compile and run the samples.

---

## Understanding the Code

Each Java sample file contains header documentation at the class level explaining the functionality or usage scenario that the sample demonstrates. You only need to edit user credentials and WSDL location information, as applicable.

Only the CNM Web service code snippets of the use case are covered in the samples. For each sample, **CommonUtil** class contains the actual code used for accessing the Web service.

## Compiling and Running Samples Using Ant Tasks

1. Navigate to the CNM Web service Java sample directories, for example **JavaSamples\CrmlIntegration**.
2. Locate the batch file **sample.bat** and the Ant configuration file **build.properties**.
3. Edit the files to ensure they reflect the values appropriate for your CNM environment, specifically, the locations of JDK 1.6 home directory and the WSDL document.

---

**Note** If the **build.properties** and **sample.bat** files do not reflect the values appropriate for your CNM environment, you will not be able to generate stubs.

---

4. Run **sample.bat**.

Running the batch file 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, it implies the build process was completed without any errors. On the contrary, if the message '**BUILD FAILED**' is displayed on the console, the configuration details in the build.properties file may be incorrect and you should verify the same.

## Appendix B

# Code Tables

The code tables in this section indicate types of class, operation states, team member, references, and errors.

*Class Identifier Code* indicates the code of the class identifier type, using a set of enumerated values to denote the class types.

<b>ClassIdentifierCode.IDEA</b>	This type code indicates that the class is an Idea.
<b>ClassIdentifierCode.QUOTE</b>	This type code indicates that the class is a Quote.
<b>ClassIdentifierCode.REQUIREMENT</b>	This type code indicates that the class is a Requirement.

*Response Status Code* indicates the results of executing the Web service operation, denoting either successful execution or any of the failed states.

<b>ResponseStatusCode.SUCCESS</b>	This type code indicates that the operation result is Success.
<b>ResponseStatusCode.FAILURE</b>	This type code indicates that the operation result is Failure.
<b>ResponseStatusCode.PARTIAL_SUCCESS</b>	This type code indicates that the operation result is Partial Success.
<b>ResponseStatusCode.WARNING</b>	This type code indicates that the operation result is Warning.

*Team Member Identifier Type Code* indicates the code of the team member identifier type, using a set of enumerated values to denote the team member types.

<b>TeamMemberIdentifierTypeCode.USER</b>	This type code indicates that the team member type is User.
<b>TeamMemberIdentifierTypeCode.GROUP</b>	This type code indicates that the team member type is Group.

*Reference Type Identifier Code* indicates the code of the reference identifier type, using a set of enumerated values to denote the reference types.

<b>ReferenceTypeIdentifierCode.URL</b>	This type code indicates that the reference type is URL.
<b>ReferenceTypeIdentifierCode.AGILEPLM</b>	This type code indicates that the reference type is Agile PLM object.
<b>ReferenceTypeIdentifierCode.FILE</b>	This type code indicates that the reference type is File.

**Error codes**

Error ID	Error Message	Comments
21000		This error message is generated by the server. It is not specific to web services.
21018	The maximum number of characters was exceeded for attribute {2}.	{2} = name of the attribute
21024	Attribute {0} is required.	{0} = name of the attribute
21036	Failed to add reference because you do not have the requisite privilege in Agile PLM.	
21037	A reference back to {0} could not be added in Agile PLM. Please check with your Agile Administrator.	{0} = name of the CNM object
21040	The maximum length of the file description is 4000 characters.	
21043	Please enter a valid URL to this action.	
21057	The input value {0} is an invalid number.	{0} = input value of the number field
21058	The input value {0} is an invalid date.	{0} = input value of the date field
21061	The value of Class Identifier is invalid.	
21062	The value of Object Number is invalid.	
21063	The value of Note is invalid.	
21064	The value of Comment is invalid.	
21065	The value of URL is invalid.	
21066	The value of URL is invalid.	
21067	The value of Object Name is too long.	
21068	The value of Status is invalid.	
21069	A reference back to this CNM object could not be added to the Agile PLM system. Please check with your Agile administrator.	
21070	The value of this Agile PLM object's Base Class (API Name) is invalid.	
21071	The value of this Agile PLM object's Object Number is invalid.	
21072	This Agile PLM object could not be found.	
21073	The value of Note Identifier is invalid.	
21074	The requested object cannot be found. The object may have been deleted.	
21075	Your access to this object is denied because	

	your role does not allow this action.	
21076	Your access is denied. You are not a member of this object.	
21077	The current User account is inactive, please contact your CNM administrator.	
21078	The current User account does not exist, please contact your CNM administrator.	
21079	The value of Description is too long.	
21080	The value of {0} is too long.	{0} = name of the attribute
21081	Attribute {0} is not modifiable.	{0} = name of the attribute
21082	This operation is not available because your authentication provider is LDAP.	
21083	The value of one of the entries in Tags is too long.	
21084	The value of {0} is invalid.	{0} = name of the attribute
21085	Attribute {0} is not visible.	{0} = name of the attribute

