

MS-DWSS Test Suite Specification

**Abstract:** This document provides information about how to configure the test suite and how the MS-DWSS test suite is designed to test the MS-DWSS Open Specification usability and accuracy. It describes test assumptions, scope and constraints. It also specifies test scenarios, detail test cases, test suite architecture and adapter design.

Contents

[1 Configuring the test suite 4](#_Toc372278982)

[1.1 Configuring the test suite client 4](#_Toc372278983)

[1.1.1 Configuring the test suite client manually 4](#_Toc372278984)

[1.1.2 Configuring the test suite client by scripts 5](#_Toc372278985)

[1.2 Configuring the system under test (SUT) 5](#_Toc372278986)

[1.2.1 Configuring the SUT manually 5](#_Toc372278987)

[1.2.2 Configuring the SUT by scripts 5](#_Toc372278988)

[1.3 Configuring the SHOULD/MAY requirements 5](#_Toc372278989)

[2 Test Suite Design 7](#_Toc372278990)

[2.1 Assumptions, scope and constraints 7](#_Toc372278991)

[Assumptions 7](#_Toc372278992)

[Scope 7](#_Toc372278993)

[In scope 7](#_Toc372278994)

[Out of scope 7](#_Toc372278995)

[Constraints 7](#_Toc372278996)

[2.2 Test suite architecture 7](#_Toc372278997)

[2.3 Technical dependencies and considerations 8](#_Toc372278998)

[Dependencies 8](#_Toc372278999)

[Encryption consideration 8](#_Toc372279000)

[2.4 Adapter design 8](#_Toc372279001)

[2.4.1 Adapter overview 8](#_Toc372279002)

[Protocol adapter 8](#_Toc372279003)

[SUT control adapter 9](#_Toc372279004)

[2.4.2 Technical feasibility of adapter approach 9](#_Toc372279005)

[Message generation 9](#_Toc372279006)

[Message consumption 9](#_Toc372279007)

[SUT control adapter 9](#_Toc372279008)

[2.4.3 Adapter abstraction layer 9](#_Toc372279009)

[Protocol adapter 9](#_Toc372279010)

[MS-DWSS Protocol adapter Interface 9](#_Toc372279011)

[SUT control adapter 10](#_Toc372279012)

[SUT control adapter interface 10](#_Toc372279013)

[2.4.4 Adapter details 10](#_Toc372279014)

[*2.4.4.1* Protocol adapter 10](#_Toc372279015)

[2.4.4.1.1 MS-DWSS protocol adapter 10](#_Toc372279016)

[Adapter interface 11](#_Toc372279017)

[Adapter implementation 11](#_Toc372279018)

[Other class 11](#_Toc372279019)

[Enumerations 11](#_Toc372279020)

[*2.4.4.2* SUT control adapter 12](#_Toc372279021)

[2.4.4.2.1 MS-DWSS SUT control adapter 12](#_Toc372279022)

[Adapter interface 12](#_Toc372279023)

[Adapter implementation 12](#_Toc372279024)

[2.5 Test Scenarios 13](#_Toc372279025)

[2.5.1 S01\_ManageSites 13](#_Toc372279026)

[Description 13](#_Toc372279027)

[Operations 13](#_Toc372279028)

[Prerequisites 13](#_Toc372279029)

[Cleanup 13](#_Toc372279030)

[2.5.2 S02\_ManageData 13](#_Toc372279031)

[Description 13](#_Toc372279032)

[Operations 13](#_Toc372279033)

[Prerequisites 14](#_Toc372279034)

[Cleanup 14](#_Toc372279035)

[2.5.3 S03\_ManageFolders 14](#_Toc372279036)

[Description 14](#_Toc372279037)

[Operations 14](#_Toc372279038)

[Prerequisites 14](#_Toc372279039)

[Cleanup 14](#_Toc372279040)

[2.5.4 S04\_ManageDocuments 14](#_Toc372279041)

[Description 14](#_Toc372279042)

[Operations 14](#_Toc372279043)

[Prerequisites 15](#_Toc372279044)

[Cleanup 15](#_Toc372279045)

[2.5.5 S05\_ManageSiteUsers 15](#_Toc372279046)

[Description 15](#_Toc372279047)

[Operations 15](#_Toc372279048)

[Prerequisites 15](#_Toc372279049)

[Cleanup 15](#_Toc372279050)

[2.6 Test case design 16](#_Toc372279051)

[2.6.1 Traditional test case design 16](#_Toc372279052)

[2.6.2 Test case description 18](#_Toc372279053)

# Configuring the test suite

## Configuring the test suite client

### Configuring the test suite client manually

Before you run the test suite, update the values in the MS-DWSS\_TestSuite.deployment.ptfconfig file. The MS-DWSS\_TestSuite.deployment.ptfconfig file can also be configured by running the client setup script.

1. Open MS-DWSS\TestSuite\MS-DWSS\_TestSuite.deployment.ptfconfig file.
2. Update the following value to specify the common configuration file.

Property name="CommonConfigurationFileName" value="SharePointCommonConfiguration.deployment.ptfconfig"

**Note**   This property can be removed or set to empty if the required properties are copied to the test suite specific configuration file. Any other changes to this property will cause all test cases in the test suite to fail during execution. The test suite first search through its specific configuration file and use the properties from there if they are defined, before looking for them from the common configuration file (if specified).

1. Update the following properties' values to match SUT settings and configuration.

* Property name="ReaderRoleUser" value="MSDWSS\_ReaderRole"
* Property name="ReaderRoleUserPassword" value="Password01!"
* Property name="NoneRoleUser" value="MSDWSS\_NoneRole"
* Property name="NoneRoleUserPassword" value="Password01!"
* Property name="RegisteredUsersEmail" value="administrator@contoso.com"
* Property name="TestDWSSWebSite" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/MSDWSS\_Site/\_vti\_bin/dws.asmx"
* Property name="InheritPermissionSite" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/MSDWSS\_InheritPermissionSite/\_vti\_bin/dws.asmx"
* Property name="SiteCollectionName" value="MSDWSS\_SiteCollection"
* Property name="SiteCollectionWithoutSubSite" value="[TransportType]://[SutComputerName]/sites/MSDWSS\_SiteCollection\_DocumentWorkspace/\_vti\_bin/dws.asmx"
* Property name="ValidTitle" value="MSDWSS\_TestTitle"
* Property name="ValidDocumentLibraryName" value="MSDWSS\_DocumentLibrary"
* Property name="ValidFolderName" value="MSDWSS\_TestFolder"
* Property name="DocumentsName" value="MSDWSS\_TestData"
* Property name="ValidFolderUrl" value="[ValidDocumentLibraryName]/[ValidFolderName]"
* Property name="ValidDocumentUrl" value="[ValidDocumentLibraryName]/[ValidFolderName]/[DocumentsName].txt"
* <Property name="SiteCollection" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/\_vti\_bin/dws.asmx" />
* Property name="NonExistentDWSSWebSiteUrl" value="[TransportType]://[SutComputerName]/NonExistentDWSSWebSiteUrl/\_vti\_bin/dws.asmx"
* Property name="NewFolderUrl" value="[ValidDocumentLibraryName]/NewFolder"
* Property name="TestDWSSSuffix" value="/\_vti\_bin/dws.asmx"
* Property name="LISTSSuffix" value="/\_vti\_bin/lists.asmx"

### Configuring the test suite client by scripts

To configure the test suite client using scripts, see section 5.2.4 of the [SharePointTestSuiteDeploymentGuide.docx](../SharePointTestSuiteDeploymentGuide.docx)

## Configuring the system under test (SUT)

### Configuring the SUT manually

To manually configure the SUT, see section 5.1.3 of the [SharePointTestSuiteDeploymentGuide.docx](../SharePointTestSuiteDeploymentGuide.docx).

### Configuring the SUT by scripts

To configure the SUT using scripts, see section 5.1.2 of the [SharePointTestSuiteDeploymentGuide.docx](../SharePointTestSuiteDeploymentGuide.docx)

## Configuring the SHOULD/MAY requirements

All the implementation of the SHOULD/MAY and endnote related requirements are pre-configured in the format "<Property name="RXXXEnabled" value="XXXX"/>" for six Microsoft product versions in six SHOULD/MAY PTFConfig files:

* MS-DWSS\_WindowsSharePointServices3\_SHOULDMAY.deployment.ptfconfig
* MS-DWSS\_SharePointServer2007\_SHOULDMAY.deployment.ptfconfig
* MS-DWSS\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig
* MS-DWSS\_SharePointServer2010\_SHOULDMAY.deployment.ptfconfig
* MS-DWSS\_SharePointFoundation2013\_SHOULDMAY.deployment.ptfconfig
* MS-DWSS\_SharePointServer2013\_SHOULDMAY.deployment.ptfconfig

If RXXXEnabled is set to true, the requirement must be checked. If false, the requirement must not be checked. For Microsoft product versions, all values should not be changed. For third-party products, the closest Microsoft product version should be chosen, and the value of RXXXEnabled should be updated according to the real product behavior. For example, if SharePoint Foundation 2010 is chosen,user can open **MS-DWSS\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig** and update the RXXXEnabled accordingly.

# Test Suite Design

## Assumptions, scope and constraints

Assumptions

None.

Scope

In scope

* This test suite will verify the accuracy and integrity of the technical content in the Open Specification against the results returned from the protocol server by using MS-DWSS 11 operations.
* This test suite will verify the Full WSDL which is provided in the Open Specification.
* This test suite will verify the server-side and testable requirements by running all the test cases on both HTTP and HTTPS.
* This test suite will verify operations over SOAP 1.1 and SOAP 1.2.

Out of scope

* This test suite will not verify the internal implementations of its transport protocol stack.
* This test suite will not verify the requirements related to client behaviors.
* This test suite will not verify the requirements related to server internal behaviors.

Constraints

None.

## Test suite architecture

This test suite verifies server-side and testable requirements obtained Open Specification. The following figure shows the architecture of this test suite.



The architecture of the test suite

The details of the MS-DWSS Test Suite architecture:

* SUT hosts the Document Workspace Web Service which test suite client runs against.
* From third-party user’s point of view, SUT is the protocol server implementation.
* The following products have been tested with the MS-DWSS test suite on Windows platform.
* Windows SharePoint Services 3.0 Service Pack 3 (SP3)
* Microsoft SharePoint Foundation 2010 Service Pack 2 (SP2)
* Microsoft SharePoint Foundation 2013 SP1
* Microsoft Office SharePoint Server 2007 Service Pack 3 (SP3)
* Microsoft SharePoint Server 2010 Service Pack 2 (SP2)
* Microsoft SharePoint Server 2013 SP1
* Test suite acts as the client to communicate with the SUT and validates the requirements gathered from MS-DWSS Open Specification.
* Test cases use the MS-DWSS adapter to call and get the results of the MS-DWSS operations.
* MS-DWSS Adapter is used in the test cases. The test cases call the methods in the interfaces to invoke the MS-DWSS operations.
* SUT control adapter is used in the test cases. The test cases call the methods in the interfaces to configure the SUT.

## Technical dependencies and considerations

Dependencies

* This test suite depends on the SOAP messaging protocol for exchanging structured data and type information.
* This test suite depends on HTTP protocol or HTTPS protocol to transmit the messages.
* This test suite depends on the wsdl.exe tool in .NET Framework SDK to generate the MS-DWSS proxy class.
* This test suite depends on the Protocol Test Framework (PTF) to derive managed adapters.
* This test suite depends on the MS-LISTSWS methods to add/delete/retrieve lists in the SUT.

Encryption consideration

* Transportation of MS-DWSS includes HTTP and HTTPS, and encryption will be handled by HTTPS.

## Adapter design

### Adapter overview

One Protocol adapter and one SUT control adapter will be designed for this test suite.

Protocol adapter

* MS-DWSS protocol adapter
* The MS-DWSS protocol adapter is a managed adapter which is derived from the ManagedAdapterBase class in the Protocol Test Framework (PTF).
* The MS-DWSS protocol adapter has the following functionalities:
* Choose HTTP or HTTPS and SOAP 1.1 or 1.2 for transport;
* Construct requests of 11 MS-DWSS operations;
* Communicate with the SUT by sending requests to the SUT and receive the corresponding responses from the SUT;
* Parse the response messages and validate the messages according to the WSDL schema;
* Generate the result log.
* The MS-DWSS protocol adapter uses the C# proxy class, which is generated by running the wsdl.exe tool against the full WSDL of this protocol to send SOAP request messages and receive SOAP response messages. The wsdl.exe can be found in Microsoft .NET Framework SDK tools.

SUT control adapter

* SUT control adapter
* The SUT control adapter is a managed adapter, which is derived from the ManagedAdapterBase class in PTF.
* The SUT control adapter has the following functionalities.
* Delete lists on the SUT.
* Add lists on the SUT.
* The SUT control adapter uses the proxy class in the MS-LISTSWS test suite to send request SOAP messages and receive SOAP response messages.
* The SUT control adapter is invoked by the test cases.

### Technical feasibility of adapter approach

Message generation

The MS-DWSS Protocol adapter gets the parameter values of the WSDL operations and calls the corresponding operations in MS-DWSS proxy class. The MS-DWSS proxy class serializes the parameter values to XML elements to format the SOAP request messages, and then the SOAP request messages are sent out by the MS-DWSS proxy class.

Message consumption

The messages received from the SUT will be parsed in the MS-DWSS proxy class and be passed upon to the MS-DWSS Protocol adapter. Then these messages are consumed in the MS-DWSS protocol adapter to validate the message format and to validate the logic-related requirements in the test cases.

SUT control adapter

The SUT control adapter are designed to remotely control the SUT to set up and clean up the test environment of the SUT and to retrieve file version information from the SUT.

### Adapter abstraction layer

Protocol adapter

MS-DWSS Protocol adapter Interface

There are 11 methods declared in the MS-DWSS adapter interface IMS\_DWSSAdapter.

The methods are described in the following table:

|  |  |  |
| --- | --- | --- |
| No. | Methods | Description |
| 1 | CanCreateDwsUrl | Determine if a client has sufficient permission rights to create a Document Workspace site at the specified URL. |
| 2 | CreateDws | Create a Document Workspace site and optionally adds users to the new SharePoint site. |
| 3 | CreateFolder | Create a subfolder in the document library of the current Document Workspace site. |
| 4 | DeleteDws | Delete the current Document Workspace site and its contents. |
| 5 | DeleteFolder | Delete a subfolder from a document library of the current Document Workspace site. |
| 6 | FindDwsDoc | Return the absolute URL of a document listed in the documents parameter of a previous call to the CreateDws method. |
| 7 | GetDwsData | Return information about a Document Workspace site and the lists it contains. |
| 8 | GetDwsMetaData | Return information about a Document Workspace site and the lists it contains. |
| 9 | RemoveDwsUser | Remove the specified user from the list of users for the current Document Workspace site. |
| 10 | RenameDws | Change the title of the current Document Workspace site. |
| 11 | UpdateDwsData | Perform a batch update of items and data for the Document Workspace site. |

Adapter interface methods

SUT control adapter

SUT control adapter interface

* There are two methods declared in the SUT control adapter interface IMS\_DWSSSUTControlAdapter.
* The two MS-LISTSWS operations: AddList, DeleteList. The operators of the two methods are abstracted the same as the operations described in the MS-LISTSWS.
* The following table shows the details of the 1 interface method.

|  |  |
| --- | --- |
| Interface Method Name | Description |
| AddList | This method is used to add the specified list in site. |
| DeleteList | This method is used to delete the specified list in site. |

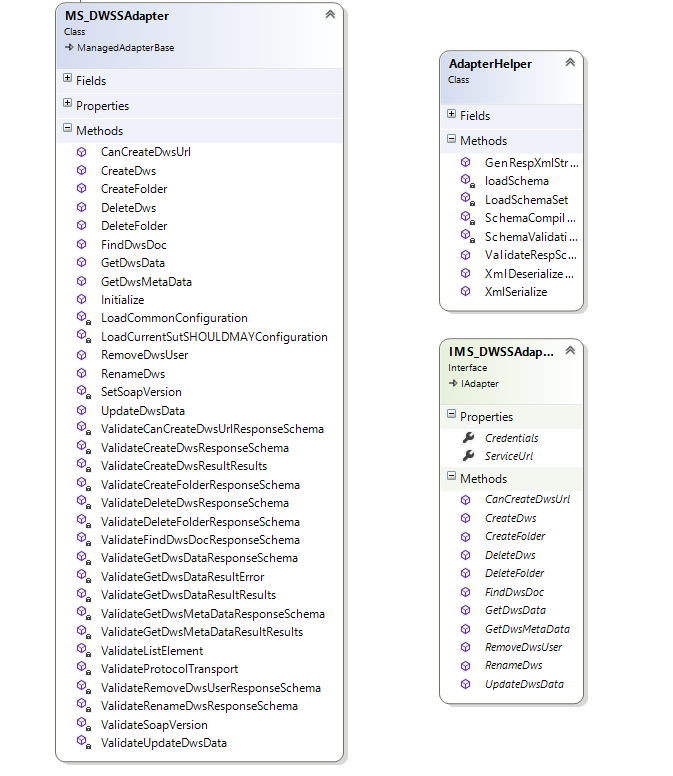
SUT control adapter methods

### Adapter details

#### Protocol adapter

##### MS-DWSS protocol adapter

The following figure shows the class diagram of the MS-DWSS adapter.



Protocol adapter class diagram

The following outlines details of the class diagram:

Adapter interface

* IMS\_DWSSAdapter is the interface of the protocol adapter.
* This interface states the methods for the corresponding MS-DWSS operations and other methods as described in [section 2.4.3 Adapter Abstract Level](#adaptermethod).

Adapter implementation

* MS\_DWSSAdapter is the protocol adapter class of the test suite. It is used to implement IMS\_DWSSAdapter.

Other class

* AdapterHelper provides the assisted methods in MS\_DWSSAdapter. It assists to encapsulate and parse string, construct URLs, and maintain temporary variables.

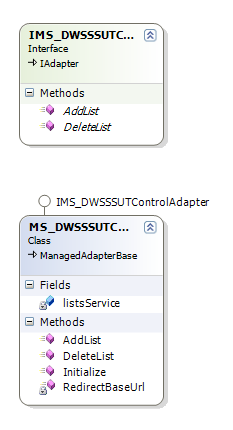
Enumerations

* ProtocolTransportType enumerates the transport type for soap messages.
* SoapVersion enumerates the allowed format types of protocol messages.

#### SUT control adapter

##### MS-DWSS SUT control adapter

The following figure shows the class diagram of the SUT control adapter.



SUT control adapter class diagram

The following outlines details of the class diagram:

Adapter interface

* IMS\_DWSSSUTControlAdapter is the interface of MS\_DWSSSUTControlAdapter.
* IMS\_DWSSSUTControlAdapter defines the methods invoked by test cases, including AddList, DeleteList method.

Adapter implementation

* MS\_DWSSSUTControlAdapter is the SUT control adapter class of the test suite. It is used to implement IMS\_DWSSSUTControlAdapter. It sends the SOAP requests and gets corresponding SOAP responses over HTTP/HTTPS.
* The AddList method is implemented by generating SOAP request, invoking the AddList methods provided by the MS-LISTSWS proxy class to send SOAP requests; getting corresponding de-serialized response.
* The DeleteList method is implemented by generating SOAP request, invoking the DeleteList methods provided by the MS-LISTSWS proxy class to send SOAP requests; getting corresponding de-serialized response.
* The Initialize method is used to initialize the SUT control adapter.
* The MS\_DWSSSUTControlAdapter is implemented by managed codes. The implementation can be substituted by other implementation or be configured as interactive mode for the third party’s need.

## Test Scenarios

Five scenarios are designed to cover in-scope and testable requirements in MS-DWSS Test Suite. The details of the scenarios are as follows:

|  |  |
| --- | --- |
| Scenario | Description |
| [S01\_ManageSites](#S1) | This scenario is designed to manage Document Workspace sites. |
| [S02\_ManageData](#S2) | This scenario is designed to manage data for the Document Workspace site. |
| [S03\_ManageFolders](#S3) | This scenario is designed to manage folders in the Document Workspace site. |
| [S04\_ManageDocuments](#S4) | This scenario is designed to manage documents in the Document Workspace site. |
| [S05\_ManageSiteUsers](#S5) | This scenario is designed to manage site users for the Document Workspace site. |

MS-DWSS scenarios

### S01\_ManageSites

Description

This scenario is designed to test add/delete/get/update operations on sites with valid or invalid Input parameters.

Operations

* CanCreateDwsUrl
* CreateDws
* RenameDws
* UpdateDwsData
* DeleteDws

Prerequisites

N/A

Cleanup

N/A

### S02\_ManageData

Description

This scenario is designed to test add/delete/get/update operations on sites data with valid or invalid Input parameters.

Operations

* CanCreateDwsUrl
* CreateDws
* GetDwsData
* GetDwsMetaData
* DeleteDws

Prerequisites

N/A

Cleanup

N/A

### S03\_ManageFolders

Description

This scenario is designed to test add/delete/get/update operations on folders with valid or invalid Input parameters.

Operations

* CanCreateDwsUrl
* CreateDws
* CreateFolder
* DeleteFolder
* DeleteDws

Prerequisites

N/A

Cleanup

N/A

### S04\_ManageDocuments

Description

This scenario is designed to test add/delete/get/update operations on documents with valid or invalid Input parameters.

Operations

* CanCreateDwsUrl
* CreateDws
* FindDwsDoc
* DeleteDws

Prerequisites

N/A

Cleanup

N/A

### S05\_ManageSiteUsers

Description

This scenario is designed to test add/delete/get/update operations on sites users with valid or invalid Input parameters.

Operations

* CanCreateDwsUrl
* CreateDws
* GetDwsData
* RemoveDwsUser
* DeleteDws

Prerequisites

N/A

Cleanup

N/A

## Test case design

### Traditional test case design

Traditional testing approach is adopted as the test approach for MS-DWSS Test Suite. The test cases are designed to cover the in-scope and testable requirements.

37 traditional test cases are designed to cover the five scenarios mentioned in [section 2.5 Test scenarios](#_Test_Scenarios). Details of test cases are described in [section 2.6.2 test case description](#_Test_Cases_Description). Test cases distributed among the scenarios are listed in the following table:

|  |  |
| --- | --- |
| Scenario ID | Test case name |
| S01\_ManageSites | [MSDWSS\_S01\_TC01\_CanCreateDwsUrl\_ValidUrl](#S1TC01) |
| [MSDWSS\_S01\_TC02\_CanCreateDwsUrl\_EmptyUrl](#S1TC02) |
| [MSDWSS\_S01\_TC03\_CanCreateDwsUrl\_HTTPError](#S1TC03) |
| [MSDWSS\_S01\_TC04\_CreateDws\_EmptyNameAndEmptyTitle](#S1TC04) |
| [MSDWSS\_S01\_TC05\_CreateDws\_EmptyNameAndValidTitle](#S1TC05) |
| [MSDWSS\_S01\_TC06\_CreateDws\_HTTPError](#S1TC06) |
| [MSDWSS\_S01\_TC07\_CreateDws\_ServerFailure](#S1TC07) |
| [MSDWSS\_S01\_TC08\_RenameDws\_ValidTitle](#S1TC08) |
| [MSDWSS\_S01\_TC09\_RenameDws\_NoAccess](#S1TC09) |
| [MSDWSS\_S01\_TC10\_DeleteDws\_DeleteCurrentSiteSuccessfully](#S1TC10) |
| [MSDWSS\_S01\_TC11\_DeleteDws\_NotExist](#S1TC11) |
| [MSDWSS\_S01\_TC12\_DeleteDws\_WebContainsSubwebs](#S1TC12) |
| [MSDWSS\_S01\_TC13\_DeleteDws\_NoAccess](#S1TC13) |
| S02\_ManageData | [MSDWSS\_S02\_TC01\_GetDwsData\_NoChanges](#S2TC01) |
| [MSDWSS\_S02\_TC02\_GetDwsData\_ListNotFound](#S2TC02) |
| [MSDWSS\_S02\_TC03\_GetDwsData\_NoAccess](#S2TC03) |
| [MSDWSS\_S02\_TC04\_GetDwsMetaData\_DocumentNotFound](#S2TC04) |
|  | [MSDWSS\_S02\_TC05\_GetDwsMetaData\_MinimalIsTrue](#S2TC05) |
| [MSDWSS\_S02\_TC06\_GetDwsMetaData\_MinimalIsFalse](#S2TC06) |
| [MSDWSS\_S02\_TC07\_GetDwsMetaData\_InheritPermission](#S2TC07) |
| [MSDWSS\_S02\_TC08\_GetDwsMetaData\_InvalidDocumentUrl](#S2TC08) |
| [MSDWSS\_S02\_TC09\_GetDwsMetaData\_WorkspaceTypeEmpty](#S2TC09) |
| [MSDWSS\_S02\_TC10\_GetDwsMetaData\_IDIsEmpty](#S2TC10) |
| [MSDWSS\_S02\_TC11\_GetDwsMetaData\_HTTPError](#S2TC11) |
| [MSDWSS\_S02\_TC12\_UpdateDwsData\_ServerFailure](#S2TC12) |
| S03\_ManageFolders | [MSDWSS\_S03\_TC01\_CreateFolder\_CreateFolderSuccessfully](#S3TC01) |
| [MSDWSS\_S03\_TC02\_CreateFolder\_FolderNotFound](#S3TC02) |
| [MSDWSS\_S03\_TC03\_CreateFolder\_AlreadyExists](#S3TC03) |
| [MSDWSS\_S03\_TC04\_CreateFolder\_NoAccess](#S3TC04) |
| [MSDWSS\_S03\_TC05\_CreateFolder\_ServerFailure](#S3TC05) |
| [MSDWSS\_S03\_TC06\_CreateFolder\_InvalidParentFolderUrl](#S3TC06) |
| [MSDWSS\_S03\_TC07\_DeleteFolder\_DeleteFolderSuccessfully](#S3TC07) |
| [MSDWSS\_S03\_TC08\_DeleteFolder\_FolderNotFound](#S3TC08) |
| S04\_ManageDocuments | [MSDWSS\_S04\_TC01\_FindDwsDoc\_ValidId](#S4TC01) |
| [MSDWSS\_S04\_TC02\_FindDwsDoc\_ItemNotFound](#S4TC02) |
| S05\_ManageSiteUsers | [MSDWSS\_S05\_TC01\_RemoveDwsUser\_RemoveUserSuccessfully](#S5TC01) |
| [MSDWSS\_S05\_TC02\_RemoveDwsUser\_ServerFailure](#S5TC02) |
| [MSDWSS\_S05\_TC03\_RemoveDwsUser\_NoAccess](#S5TC03) |

Test case scenario distribution

* The successful test cases are designed to verify the Soap Out response messages and the core operations of this protocol. For example, the test case [MSDWSS\_S01\_TC01\_CanCreateDwsUrl\_ValidaUrl](#S1TC01).
* Negative Testing: in order to get the Error Type mentioned in Open specification, we applied this testing technique. For example, the test case [MSDWSS\_S01\_TC03\_CanCreateDwsUrl\_HTTPError](#S1TC03).

### Test case description

MS-DWSS Test Suite uses Traditional Testing as the test approach. The test case design is based on the Protocol adapter interfaces defined in [section 2.4.3](#_Adapter_Abstract_LevelAdapter) and test scenarios defined in [section 2.5](#_Test_Scenarios).

* The steps in each test case definition use methods and parameter names in the adapter interfaces directly.
* All the test cases use HTTP and HTTPS as transport protocol.

The following tables describe the details of 37 traditional test cases.

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC01\_CanCreateDwsUrl\_ValidUrl |
| Description | This test case is intended to validate the result node returned by CanCreateDwsUrl operation when using the valid URL. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CanCreateDwsUrl method with a valid relative URL to get a proposed Document Workspace URL. 2. Call CreateDws method to create a new Document Workspace using the proposed URL to verify that the proposed URL returned by CanCreateDwsUrl is valid.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call DeleteDws method to delete the Document Workspace created on step 2. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC01\_CanCreateDwsUrl\_ValidaUrl

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC02\_CanCreateDwsUrl\_EmptyUrl |
| Description | This test case is intended to validate the result node returned by CanCreateDwsUrl operation when using the empty URL. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CanCreateDwsUrl method with an empty relative URL to get a new web site name. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC02\_CanCreateDwsUrl\_EmptyAndInvalidUrl

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC03\_CanCreateDwsUrl\_HTTPError |
| Description | This test case is intended to validate the result node that contains HTTP Error returned by CanCreateDwsUrl operation when the user is not authorized. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CanCreateDwsUrl method with a valid relative URL to get a proposed Document Workspace URL. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC03\_CanCreateDwsUrl\_HTTPError

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC04\_CreateDws\_EmptyNameAndEmptyTitle |
| Description | This test case is intended to validate the result node returned by CreateDws operation using the empty name and empty title. |
| Prerequisites | N/A. |
| Test execution steps | 1. 1. Call CreateDws method to create a new Document Workspace. 2. **Input parameters:**  * dwsName: empty name of the Document Workspace site * users: invalid users to be added as contributors in the Document Workspace site * dwsTitle: empty title for the workspace * docs: no document to be added in the dws.  1. Call GetDwsData method to get the title of Document Workspace.   **Input parameters:**   * docUrl: a valid site-based URL of a document in the document library in a document workspace. * lastUpdate: contains the empty string.  1. Call DeleteDws method to delete the Document Workspace created on step 2. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC04\_CreateDws\_EmptyNameAndEmptyTitle

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC05\_CreateDws\_EmptyNameAndValidTitle |
| Description | This test case is intended to validate the result node returned by CreateDws operation using empty name and valid title. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create two new Document Workspaces with the same title.   **Input parameters:**   * dwsName: empty name of the Document Workspace site * users: invalid users to be added as contributors in the Document Workspace site * dwsTitle: empty title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call DeleteDws method to delete the Document Workspaces created on step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC05\_CreateDws\_EmptyNameAndValidTitle

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC06\_CreateDws\_HTTPError |
| Description | This test case is intended to validate the result node that contains HTTP Error returned by CreateDws operation when the user is not authorized to create the specified Document Workspace. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC06\_CreateDws\_HTTPError

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC07\_CreateDws\_ServerFailure |
| Description | This test case is intended to validate the result node that contains ServerFailure Error returned by CreateDws operation when using same input parameters twice to create the Document Workspace. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: docs to be added in the Document Workspace site  1. Call CreateDws method again with the same Input parameters as step 2 to trigger a ServerFailure error.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: docs to be added in the Document Workspace site  1. Call DeleteDws method to delete the Document Workspaces created on step 1, 2. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC07\_CreateDws\_ServerFailure

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC08\_RenameDws\_ValidTitle |
| Description | This test case is intended to validate the result node returned by RenameDws operation when using a valid title for the Document Workspace site. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsName: empty name of the Document Workspace site * users: users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: docs to be added in the Document Workspace site  1. Call RenameDws method to rename the title of the Document Workspace.   **Input parameters:**   * dwsTitle: a string that contains the new title for the workspace  1. Call GetDwsData method to retrieve the general information about the Document Workspace site.   **Input parameters:**   * docUrl: a site-relative URL that specifies the list or document * lastUpdate: the empty update type of Dws data  1. Call RenameDws method to rename Dws without title.   **Input parameters:**   * dwsTitle: no title for the workspace  1. Call DeleteDws method to delete the Document Workspace created on step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC08\_RenameDws\_ValidTitle

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC09\_RenameDws\_NoAccess |
| Description | This test case is intended to validate the result node returned by RenameDws operation when the user does not have sufficient access to rename the title for the Document Workspace site. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsName: empty name of the Document Workspace site * users: users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: docs to be added in the Document Workspace site  1. Call RenameDws method by the user having no permission.   **Input parameters:**   * dwsTitle: a valid title for the Document Workspace  1. Call DeleteDws method with an administrator that has sufficient permission to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC09\_RenameDws\_NoAccess

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC10\_DeleteDws\_DeleteCurrentSiteSuccessfully |
| Description | This test case is intended to validate the result node returned by DeleteDws operation when successfully deleting the current Document Workspace site and its contents. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call DeleteDws method to delete the Document Workspace created on step 2. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC10\_DeleteDws\_DeleteCurrentWebSiteSuccessfully

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC11\_DeleteDws\_NotExist |
| Description | This test case is intended to validate the result node returned by DeleteDws operation when the target Document Workspace does not exist. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call DeleteDws method to delete the Document Workspace that does not exist. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC11\_DeleteDws\_DeleteDws\_NotExist

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC12\_DeleteDws\_WebContainsSubwebs |
| Description | This test case is intended to validate the result node that contains WebContainsSubwebs error returned by DeleteDws operation. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsTitle: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call CreateDws method to create a Document Workspace sub-site.   **Input parameters:**   * dwsTitle: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call DeleteDws method to delete the Document Workspace created on step 1 which contains the sub-site. 2. Call DeleteDws method to delete the Document Workspace created on step 5 which contains the sub-site. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC12\_DeleteDws\_WebContainsSubwebs

|  |  |
| --- | --- |
| S01\_ManageSites | |
| Test case ID | MSDWSS\_S01\_TC13\_DeleteDws\_NoAccess |
| Description | This test case is intended to validate the result node returned by DeleteDws operation when the user does not have sufficient access to delete the target Document Workspace site. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsTitle: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call DeleteDws method by the user having no sufficient rights. 2. Call DeleteDws method to delete the site collection. 3. Call DeleteDws method to delete the site created in step 1 with sufficient rights. |
| Cleanup | Common Cleanup. |

MSDWSS\_S01\_TC13\_DeleteDws\_NoAccess

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC01\_GetDwsData\_NoChanges |
| Description | This test case is intended to validate that if the list in the Document Workspace has not changed since lastUpdate, GetDwsData must return a NoChanges child element of the List element as specified in GetDwsDataResponse. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsData method to retrieve the general information about the Document Workspace site with valid document URL.   **Input parameters:**   * docUrl: a valid site-based URL of a document in the document library in a Document Workspace * lastUpdate: contains the empty lastUpdate value returned in the result  1. Call GetDwsData method again to retrieve the general information about the Document Workspace site without any update.   **Input parameters:**   * docUrl: a valid site-based URL of a document in the document library in a Document Workspace * lastUpdate: contains the lastUpdate value returned in the result of a previous call to GetDwsData  1. Call GetDwsData method to retrieve the general information about the Document Workspace site without document URL and lastUpdate.   **Input parameters:**   * docUrl: the site-relative URL is null * lastUpdate: the lastUpdate is null |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC01\_GetDwsData\_NoChanges

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC02\_GetDwsData\_ListNotFound |
| Description | This test case is intended to validate that an Error element with ListNotFound code should be returned if the server cannot locate the document. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsData method to retrieve the general information about the Document Workspace site.   **Input parameters:**   * docUrl: an invalid document URL * lastUpdate: an empty string |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC02\_GetDwsData\_ListNotFound

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC03\_GetDwsData\_NoAccess |
| Description | This test case is intended to validate that an Error element with NoAccess code should be returned if the server detects an access restriction during processing. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsData method to retrieve the general information about the Document Workspace.   **Input parameters:**   * docUrl: a valid document URL * lastUpdate: empty string |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC03\_GetDwsData\_NoAccess

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC04\_GetDwsMetaData\_DocumentNotFound |
| Description | This test case is intended to validate that an Error element with DocumentNotFound code should be returned if the server cannot find the document. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsTitle: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call CreateFolder method to create a subfolder in the Document Workspace.   **Input parameters:**   * folderUrl: a valid folder URL  1. Call GetDwsMetaData method to get the metadata of the DWS; however, no document exists under the subfolder in the Document Workspace.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: the ID of purpose document * isMinimal: false indicating the output includes extra information |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC04\_GetDwsMetaData\_DocumentNotFound

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC05\_GetDwsMetaData\_MinimalIsTrue |
| Description | This test case is intended to validate the result node returned by GetDwsMetaData operation when the parameter minimal is true. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsMetaData method to get the metadata.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: a valid GUID * isMinimal: true indicating the output only includes minimal information. |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC05\_GetDwsMetaData\_MinimalIsTrue

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC06\_GetDwsMetaData\_MinimalIsFalse |
| Description | This test case is intended to validate the result node returned by GetDwsMetaData operation when the parameter minimal is false. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsMetaData method get the metadata.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: a GUID that a client uses instead of the document URL * isMinimal: false indicating the output includes extra information  1. Call CreateDws method to create a Document Workspace.   **Input parameters:**   * dwsTitle: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call AddList method to add three lists in the Document Workspace.   **Input parameters:**   * listName: The name of the list to be added. * templateID: The template id of the new list. * baseUrl: The service URL of the current adapter instance.  1. Call GetDwsMetaData method get the metadata.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: a GUID that a client uses instead of the document URL * isMinimal: false indicating the output includes extra information  1. Call DeleteList method to delete the lists in the Document Workspace.   **Input parameters:**   * listName: The name of the list to be deleted. * baseUrl: The service URL of the current adapter instance.  1. Call GetDwsMetaData method get the metadata.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: a GUID that a client uses instead of the document URL * isMinimal: false indicating the output includes extra information |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC06\_GetDwsMetaData\_MinimalIsFalse

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC07\_GetDwsMetaData\_InheritPermission |
| Description | This test case is intended to validate the result node returned by GetDwsMetaData operation when the inherited site is created. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsMetaData method to get metadata of Dws.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: the ID of purpose document * isMinimal: true indicating the output only includes minimal information. |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC07\_GetDwsMetaData\_InheritPermission

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC08\_GetDwsMetaData\_InvalidDocumentUrl |
| Description | This test case is intended to validate the result node returned by GetDwsMetaData operation when the invalid document URL is request. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsMetaData method to get the metadata of Dws with invalid document URL.   **Input parameters:**   * docUrl: an invalid document URL * docID: a valid document ID * isMinimal: true indicating the output only includes minimal information |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC08\_GetDwsMetaData\_InvalidDocumentUrl

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC09\_GetDwsMetaData\_WorkspaceTypeEmpty |
| Description | This test case is intended to validate the result node returned by GetDwsMetaData operation when the site is the site collection, whose workspace type returned is not DWS or MWS. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsMetaData method to get the metadata of Dws with valid document URL.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: a GUID that a client uses instead of the document URL * isMinimal: true indicating the output only includes minimal information |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC09\_GetDwsMetaData\_WorkspaceTypeEmpty

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC10\_GetDwsMetaData\_IDIsEmpty |
| Description | This test case is intended to validate the result node returned by GetDwsMetaData operation when the parameter id is empty. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call GetDwsMetaData method to get the metadata of Dws with document URL and document ID.   **Input parameters:**   * docUrl: a valid site-relative document URL. * docID: an empty document ID * isMinimal: false indicating the output includes extra information  1. Call GetDwsMetaData method to get the metadata of Dws without document URL.   **Input parameters:**   * docUrl: Do not specify the document URL. * docID: an empty document ID * isMinimal: false indicating the output includes extra information  1. Call GetDwsMetaData method to get the metadata of Dws without document URL and document ID.   **Input parameters:**   * docUrl: Do not specify the document URL. * docID: Do not specify the document ID * isMinimal: false indicating the output includes extra information |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC10\_GetDwsMetaData\_IDIsEmpty

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC11\_GetDwsMetaData\_HTTPError |
| Description | This test case is intended to validate the result node that contains HTTP Error returned by GetDwsMetaData operation when the authenticated user is not permitted to access this information. |
| Prerequisites | N/A. |
| Test execution steps | * + - 1. Call GetDwsMetaData method to get the metadata of Dws without document URL and document ID.   **Input parameters:**   * docUrl: a valid site-relative URL which specifies the list or document * docID: a GUID that a client uses instead of the document URL * isMinimal: false indicating the output includes extra information |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC11\_GetDwsMetaData\_HTTPError

|  |  |
| --- | --- |
| S02\_ManageData | |
| Test case ID | MSDWSS\_S02\_TC12\_UpdateDwsData\_ServerFailure |
| Description | This test case is intended to validate that an Error element with ServerFailure code should be returned when the UpdateDwsData operation is failed. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call UpdateDwsData method to update Dws data with updates and meetingInstance set to empty string.   **Input parameters:**   * updates: an empty string * meetingInstance: an empty string  1. Call UpdateDwsData method without updates and meetingInstance.   **Input parameters:**   * updates: null * meetingInstance: null  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S02\_TC12\_UpdateDwsData\_ServerFailure

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC01\_CreateFolder\_CreateFolderSuccessfully |
| Description | This test case is intended to validate the result node returned by CreateFolder operation to create a subfolder in the document library of the current Document Workspace site successfully. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call CreateFolder method to create a subfolder in the Document Workspace.   **Input parameters:**   * folderUrl: a new valid folder URL  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC01\_CreateFolder\_CreateFolderSuccessfully

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC02\_CreateFolder\_FolderNotFound |
| Description | This test case is intended to validate that an Error element with FolderNotFound code should be returned by CreateFolder operation if the server cannot locate the folder. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateFolder method with a non-existent folder URL to create a folder in the document library of the Document Workspace.   **Input parameters:**   * folderUrl: an invalid URL which has no parent folder. |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC02\_CreateFolder\_FolderNotFound

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC03\_CreateFolder\_AlreadyExists |
| Description | This test case is intended to validate that an Error element with AlreadyExists code should be returned by CreateFolder operation if the specified URL already exists. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call CreateFolder method to create a folder in the Document Workspace.   **Input parameters:**   * folderUrl: a valid folder URL  1. Call CreateFolder method to create a folder using the same folder URL as step 4.   **Input parameters:**   * folderUrl: a folder URL which has already been created by previous operation  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC03\_CreateFolder\_AlreadyExists

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC04\_CreateFolder\_NoAccess |
| Description | This test case is intended to validate that an Error element with NoAccess code should be returned by CreateFolder operation when the user does not have sufficient access permissions to create the folder. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateFolder method by the user that does not have sufficient permission to create a folder in the document library of the Document Workspace.   **Input parameters:**   * folderUrl: a valid folder URL |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC04\_CreateFolder\_NoAccess

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC05\_CreateFolder\_ServerFailure |
| Description | This test case is intended to validate that an Error element with ServerFailure code should be returned by CreateFolder operation. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateFolder method by the user that has no permission to create a folder in the document library of the Document Workspace.   **Input parameters:**   * folderUrl: a valid folder URL |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC05\_CreateFolder\_ServerFailure

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC06\_CreateFolder\_InvalidParentFolderUrl |
| Description | This test case is intended to validate the result node returned by CreateFolder operation to validate that if URL contains multiple folders and the first folder in the URL doesn't exist in the site, the server will ignore the first folder in the URL and replace it with "Shared Documents". |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateFolder method to create an invalid folder in the document library of the Document Workspace site.   **Input parameters:**   * folderUrl: an invalid folder URL |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC06\_CreateFolder\_InvalidParentFolderUrl

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC07\_DeleteFolder\_DeleteFolderSuccessfully |
| Description | This test case is intended to validate the result node returned by DeleteFolder operation while deleting a subfolder in the document library of the current Document Workspace site successfully. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call CreateFolder method to create a folder in the document library of the Document Workspace site.   **Input parameters:**   * folderUrl: a valid folder URL  1. Call DeleteFolder method to delete the folder with nonexistent folder URL.   **Input parameters:**   * folderUrl: a nonexistent folder URL  1. Call DeleteFolder method to delete the folder without URL.   **Input parameters:**   * folderUrl: no folder URL  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC07\_DeleteFolder\_DeleteFolderSuccessfully

|  |  |
| --- | --- |
| S03\_ManageFolders | |
| Test case ID | MSDWSS\_S03\_TC08\_DeleteFolder\_FolderNotFound |
| Description | This test case is intended to validate the result node returned by DeleteFolder operation while the parent folder for the specified URL does not exist. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call DeleteFolder method to delete an invalid folder from the document library on the Document Workspace.   **Input parameters:**   * folderUrl: an invalid URL which has no parent folder |
| Cleanup | Common Cleanup. |

MSDWSS\_S03\_TC08\_DeleteFolder\_FolderNotFound

|  |  |
| --- | --- |
| S04\_ManageDocuments | |
| Test case ID | MSDWSS\_S04\_TC01\_FindDwsDoc\_ValidId |
| Description | This test case is intended to validate that the returning absolute URL of a document listed in the documents parameter of a previous call to the CreateDws method successfully. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call FindDwsDoc method to obtain the URL for the Document Workspace created on step 2.   **Input parameters:**   * docID: a valid document id in the Document Workspace site  1. Call FindDwsDoc method to obtain the URL for the Document Workspace created on step 2 without document id.   **Input parameters:**   * docID: no document id in the Document Workspace site  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S04\_TC01\_FindDwsDoc\_ValidId

|  |  |
| --- | --- |
| S04\_ManageDocuments | |
| Test case ID | MSDWSS\_S04\_TC02\_FindDwsDoc\_ItemNotFound |
| Description | This test case is intended to validate that an Error element with ItemNotFound code should be returned if the server cannot locate a document from the specified document ID. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call FindDwsDoc method to obtain a URL with an invalid document id in the Document Workspace.   **Input parameters:**   * docID: an invalid document id which didn’t exist in the Document Workspace site |
| Cleanup | Common Cleanup. |

MSDWSS\_S04\_TC02\_FindDwsDoc\_ItemNotFound

|  |  |
| --- | --- |
| S05\_ManageSiteUsers | |
| Test case ID | MSDWSS\_S05\_TC01\_RemoveDwsUser\_RemoveUserSuccessfully |
| Description | This test case is intended to validate that the RemoveDwsUser operation removes the specified user from the list of users for the current Document Workspace site successfully. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * + dwsName: name of the Document Workspace site   + users: the users to be added as contributors in the Document Workspace site   + dwsTitle: title for the workspace   + docs: information to be stored as a key-value pair in the site metadata  1. Call GetDwsData method to get the user id.   **Input parameters:**   * docUrl: a site-relative URL that specifies the list or document * lastupdate: an empty string  1. Call RemoveDwsUser method to delete the user from the Document Workspace created on step 2.   **Input parameters:**   * userId: a valid user id in the Document Workspace site  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S05\_TC01\_RemoveDwsUser\_RemoveUserSuccessfully

|  |  |
| --- | --- |
| S05\_ManageSiteUsers | |
| Test case ID | MSDWSS\_S05\_TC02\_RemoveDwsUser\_ServerFailure |
| Description | This test case is intended to validate the related requirements when the server returns a ServerFailure Error element during processing RemoveDwsUser operation. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call RemoveDwsUser method to delete the user from the Document Workspace created on step 2.   **Input parameters:**   * userId: an invalid user id which didn’t exist in the Document Workspace site  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S05\_TC02\_RemoveDwsUser\_ServerFailure

|  |  |
| --- | --- |
| S05\_ManageSiteUsers | |
| Test case ID | MSDWSS\_S05\_TC03\_RemoveDwsUser\_NoAccess |
| Description | This test case is intended to validate that an Error element with NoAccess code should be returned by RemoveDwsUser operation if the user does not have sufficient access permissions to remove the user. |
| Prerequisites | N/A. |
| Test execution steps | 1. Call CreateDws method to create a new Document Workspace.   **Input parameters:**   * dwsName: name of the Document Workspace site * users: the users to be added as contributors in the Document Workspace site * dwsTitle: title for the workspace * docs: information to be stored as a key-value pair in the site metadata  1. Call GetDwsData method to get the user id.   **Input parameters:**   * docUrl: a valid site-based URL of a document in the document library in a Document Workspace * lastUpdate: contains the empty lastUpdate value returned in the result  1. Call RemoveDwsUser method to delete the user from the Document Workspace created on step 2.   **Input parameters:**   * userId: a valid user id  1. Call DeleteDws method to delete the Document Workspace created in step 1. |
| Cleanup | Common Cleanup. |

MSDWSS\_S05\_TC03\_RemoveDwsUser\_NoAccess