

MS-WWSP Test Suite Specification

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

Contents

[1 Configuring the test suite 4](#_Toc403575369)

[1.1 Configuring the test suite client 4](#_Toc403575370)

[1.1.1 Configuring the test suite client manually 4](#_Toc403575371)

[1.1.2 Configuring the test suite client by scripts 5](#_Toc403575372)

[1.2 Configuring the test suite system under test (SUT) 5](#_Toc403575373)

[1.2.1 Configuring the SUT manually 5](#_Toc403575374)

[1.2.2 Configuring the SUT by scripts 5](#_Toc403575375)

[1.3 Configuring the SHOULD/MAY requirements 5](#_Toc403575376)

[2 Test suite design 6](#_Toc403575377)

[2.1 Assumptions, scope and constraints 6](#_Toc403575378)

[Assumptions 6](#_Toc403575379)

[Scope 6](#_Toc403575380)

[In scope 6](#_Toc403575381)

[Out of scope 6](#_Toc403575382)

[Constraints 6](#_Toc403575383)

[2.2 Test suite architecture 6](#_Toc403575384)

[2.3 Technical dependencies and encryption considerations 7](#_Toc403575385)

[Dependencies 7](#_Toc403575386)

[Encryption consideration 8](#_Toc403575387)

[2.4 Adapter design 8](#_Toc403575388)

[2.4.1 Adapter overview 8](#_Toc403575389)

[Protocol adapter: 8](#_Toc403575390)

[SUT control adapter: 8](#_Toc403575391)

[2.4.2 Technical feasibility of adapter approach 9](#_Toc403575392)

[Message generation 9](#_Toc403575393)

[Message consumption 9](#_Toc403575394)

[SUT control adapter 9](#_Toc403575395)

[2.4.3 Adapter abstract layer 9](#_Toc403575396)

[Protocol adapter: 9](#_Toc403575397)

[SUT control adapters: 9](#_Toc403575398)

[2.4.4 Adapter details 10](#_Toc403575399)

[*2.4.4.1* Protocol adapter 10](#_Toc403575400)

[2.4.4.1.1 MS-WWSP protocol adapter 10](#_Toc403575401)

[*2.4.4.2* SUT control adapters 11](#_Toc403575402)

[2.4.4.2.1 SUT control adapter class diagram 11](#_Toc403575403)

[2.5 Test scenarios 12](#_Toc403575404)

[2.5.1 S01\_StartWorkflow 13](#_Toc403575405)

[Description 13](#_Toc403575406)

[Operations 13](#_Toc403575407)

[Prerequisites 13](#_Toc403575408)

[Cleanup 13](#_Toc403575409)

[2.5.2 S02\_GetForItem 13](#_Toc403575410)

[Description 13](#_Toc403575411)

[Operations 13](#_Toc403575412)

[Prerequisites 13](#_Toc403575413)

[Cleanup 13](#_Toc403575414)

[2.5.3 S03\_AlterToDo 13](#_Toc403575415)

[Description 13](#_Toc403575416)

[Operations 13](#_Toc403575417)

[Prerequisites 14](#_Toc403575418)

[Cleanup 14](#_Toc403575419)

[2.5.4 S04\_ClaimReleaseTask 14](#_Toc403575420)

[Description 14](#_Toc403575421)

[Operations 14](#_Toc403575422)

[Prerequisites 14](#_Toc403575423)

[Cleanup 14](#_Toc403575424)

[2.6 Test case design 15](#_Toc403575425)

[2.6.1 Traditional test case design 15](#_Toc403575426)

[2.6.2 Test case description 15](#_Toc403575427)

# 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-SITESS\_TestSuite.deployment.ptfconfig file. The MS-WWSP\_TestSuite.deployment.ptfconfig file can also be configured by running the client setup script.

1. Open MS-WWSP\TestSuite\MS-WWSP\_TestSuite.deployment.ptfconfig.
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="TargetServiceUrl" value="[TransportType]://[SUTComputerName]/sites/[SiteCollectionName]/\_vti\_bin/workflow.asmx"
* Property name="SiteCollectionName" value="MSWWSP\_SiteCollection"
* Property name="WorkflowAssociationName" value="MSWWSP\_Workflow"
* Property name="MSWWSPTestAccount" value="MSWWSP\_User"
* Property name="KeyWordForAssignedToField" value="MSWWSP\_User"
* Property name="MSWWSPTestAccountPassword" value="Password01!"
* Property name="UserGroupOnSUT" value="MSWWSP\_UserGroup"
* Property name="CurrentDocLibraryListName" value="MSWWSP\_DocumentLibrary"
* Property name="CurrentTaskListName" value="Tasks"

1. The following properties are not associated with SUT settings and can normally retain with default values.

* Property name="startworkflowParameterDataFile" value="startworkflowParameters\_[SUTVersionShortName].xml"
* Property name="startworkflowParameterDataFileForClaim" value="startworkflowParameters\_Claim\_[SUTVersionShortName].xml"
* Property name="WorkFlowAssociationXsdFile" value="workflowAssociationData\_[SUTVersionShortName].xsd"
* Property name="AlertToDoDataFile" value="AlertToDoData.xml"
* Property name="AlertedValuePlaceHolder" value="[AlertTodoUpdatedValue]"
* Property name="AssignedToPlaceHolder" value="[AssignedTo]"
* Property name="ValidateWorkFlowAssociation" value="true"
* Property name="ServiceTimeOut" value="10"
* Property name="DelayTimeBeforeModifyTask" value="0"

### 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 test suite 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

Implementation of the SHOULD/MAY and endnote-related requirements are pre-configured in the format "<Property name="RXXXEnabled" value= "XXXX"/>" for the product versions in the following config files:

* MS-WWSP\_WindowsSharePointServices3\_SHOULDMAY.deployment.ptfconfig
* MS-WWSP\_SharePointServer2007\_SHOULDMAY.deployment.ptfconfig
* MS-WWSP\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig
* MS-WWSP\_SharePointServer2010\_SHOULDMAY.deployment.ptfconfig
* MS-WWSP\_SharePointFoundation2013\_SHOULDMAY.deployment.ptfconfig
* MS-WWSP\_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-WWSP\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig** and update the RXXXEnabled accordingly.

# Test suite design

## Assumptions, scope and constraints

Assumptions

None.

Scope

In scope

* MS-WWSP test suite will verify the accuracy and integrity of the parameters and schemas of 7 operations in the Open Specification against the results returned from the protocol server by using MS-WWSP operations.
* This test suite will verify the Full WSDL which is provided in the Open Specification.
* MS-WWSP test suite will verify the testable requirements by running all the test cases on HTTP or HTTPS.
* This test suite will verify the server-side and testable requirements by running all the test cases with formatting of request and response messages in Soap 1.1 and Soap 1.2.

Out of scope

* 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.
* This test suite will not verify the internal implementations of its transport protocol stack.

Constraints

None.

## Test suite architecture

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

**The MS-WWSP test suite architecture**

The details of the MS-WWSP test suite architecture as follows:

* The SUT hosts the MS-WWSP Web Service which test suite runs against.
* From third-party user’s point of view, SUT is the protocol server implementation.
* The SUTs in this test suite are the following 3 products running on Windows platform which supports the features of MS-WWSP.
* Microsoft Office SharePoint Server 2007 SP3
* Microsoft SharePoint Server 2010 SP2
* Microsoft SharePoint Server 2013 SP1
* Test suite acts as the client to communicate with the SUT and validates the requirements gathered from MS-WWSP Open Specification.
* Test cases invoke the MS-WWSP adapter to call the MS-WWSP operations and validates the response from SUT. Test cases also use the SUT control adapter and the WWSP SUT control adapter to set the SUT to the test case specific situation.
* MS-WWSP adapter is used in the test cases. The test cases call the methods in the interfaces to invoke the MS-WWSP operations.
* WWSP SUT control adapter is used in the test cases. The test cases call the methods in the interfaces to configure the SUT.
* The test cases also use the SUT control adapter to set/modify the SUT environment by calling the methods in the interface to configure the SUT.

## Technical dependencies and encryption 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 the .NET Framework Software Development Kit (SDK) to generate the MS-WWSP proxy class.
* This test suite depends on Protocol Test Framework (PTF) to derive managed adapters.

Encryption consideration

* Transportation of MS-WWSP 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-WWSP adapter
* The MS-WWSP adapter is a managed adapter which is derived from the ManagedAdapterBase class in PTF.
* MS-WWSP adapter Interface declares the interface of the MS-WWSP Adapter.
* The MS-WWSP adapter Implementation implements the interface mentioned above. It directly uses the methods and the data structures in the MS-WWSP proxy class.
* The MS-WWSP adapter has the following functionalities:
* Choose HTTP or HTTPS and SOAP 1.1 or SOAP 1.2 for transport;
* Construct requests of 7 MS-WWSP operations;
* Communicate with the SUT by sending requests to the SUT and receiving the corresponding responses from the SUT;
* Analyze the grammar of the response messages and validate the messages according to the WSDL schema;
* Generate the result log.
* The MS-WWSP 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 will be a scripted adapter.
* SUT control adapter Interface declares the interface of the SUT control adapter.
* The SUT control adapter Implementation implements the interface mentioned above. It contains managed data structures of MS-WWSP protocol and directly invokes the web service methods which defined in .NET namespace "System.Net".
* The SUT control adapter has the following functionalities:
* Upload files into a document library on the site.
* Get information of current task list, workflow association, and web site from SUT.
* Clean up uploaded files and workflow tasks.
* The SUT control adapter is invoked by the test cases.

### Technical feasibility of adapter approach

Message generation

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

Message consumption

The messages received from the SUT will be analyzed for grammar in the MS-WWSP proxy class and be passed upon to the MS-WWSP Adapter. Then these messages are consumed in the MS-WWSP Adapter to validate the message format and to validate the logic-related requirements in the test cases.

SUT control adapter

The SUT control adapter is designed to upload files, cleanup workflow tasks, cleanup uploaded files and get some information from SUT.

### Adapter abstract layer

Protocol adapter:

* MS-WWSP adapter interface
* There are seven methods declared in the MS-WWSP adapter interface IMS\_WWSPAdapter.
* Seven of the methods correspond to the seven MS-WWSP operations. The operators of the seven methods are abstracted the same as the operations specified in the MS-WWSP.

SUT control adapters:

* SUT control adapter interface
* There are nine methods declared in the SUT control adapter interface IMSWWSPSUTControlAdapter, the following table shows the details of these nine interface methods.

|  |  |
| --- | --- |
| Interface method name | Description |
| UploadFileToDocumentLibrary | Upload files into a document library on the site |
| GetCurrentWebTitle | Get the current web title from SUT. |
| GetListIdByName | Get the list id by specified list name. |
| GetWorkflowAssociationIdByName | Get the Workflow association Id according to the workflow association name. |
| CleanUpUploadedFiles | Clean up uploaded files by specified files' URLs |
| CleanUpStartedTasks | Clean up the workflow tasks by specified task ids |
| GetListUrlByName | Get the list URL by specified list name. |
| GetCurrentWebUrl | Get the current web URL of the web site. |
| GetBaseIdOfWorkFlowAssociation | Get the Workflow association base Id according to the workflow association name. |

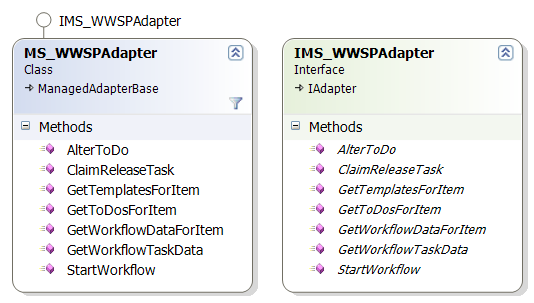
SUT control adapter interface

### Adapter details

#### Protocol adapter

##### MS-WWSP protocol adapter

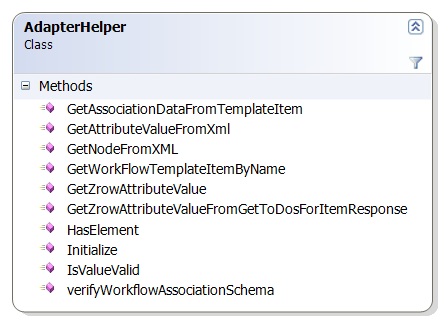
The following figure shows the class diagram of the MS-WWSP protocol adapter.



Protocol adapter

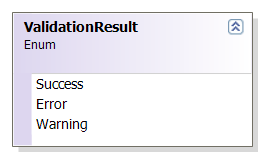
The following outlines details of the class diagram:

* Adapter interface
* IMS\_WWSPAdapter is the interface of the Protocol adapter.
* IMS\_WWSPAdapter defines the seven methods which are specified in Open Specification invoked by test cases.
* Adapter implementation
* MS\_ WWSPAdapter is the Protocol Adapter class of the test suite. It is used to implement IMS\_WWSPAdapter.
* The 7 methods defined in IMS\_WWSPAdapter are implemented by generating SOAP requests, and then invoking these methods provided by the MS-WWSP proxy class to send SOAP requests; getting corresponding de-serialized response and verifying related adapter requirements.
* The Initialize method is used to configure HTTP or HTTPS and SOAP 1.1 or SOAP 1.2 for transport.
* Other class
* WorkflowSoap is a schema validation class, which is a part of the proxy generated from the MS-WWSP WSDL. It overrides the Invoke method to generate a schema validation reader, and then use the kind of mechanism to validate whether the result xml is consistent with the XSD.
* The AdapterHelper class provides the methods of getting value from different complex type and verifying workflow association data schema, getting and verifying URL, etc.



Adapter helper

* Enumeration
* ValidationResult is used to specify the result of schema validation for request/response message.

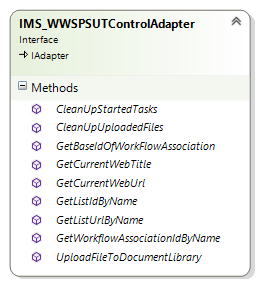


Enumeration

#### SUT control adapters

##### SUT control adapter class diagram

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
* IMSWWSPSUTControlAdapter is the interface of SUTControlAdapter.
* IMSWWSPSUTControlAdapter defines the nine methods which are invoked by test cases.
* Adapter implementation
* SUTControlAdapter is the SUT control adapter class of the test suite. It is used to implement IMSWWSPSUTControlAdapter.
* The SUTControlAdapter is implemented by Microsoft PowerShell script. The implementation can be substituted by other implementation for the third party’s need.

## Test scenarios

Four scenarios are designed to cover testable requirements in MS-WWSP test suite. The details of the scenarios are as follows:

|  |  |
| --- | --- |
| Scenario | Description |
| [S01\_StartWorkflow](#s1) | The client tries to start a workflow task for a document item by specified workflow association. |
| [S02\_GetForItem](#s2) | The client tries to get workflow related data from SUT, include workflow association data, workflow task and workflow data. |
| [S03\_AlterToDo](#s3) | The client tries to update a workflow task. |
| [S04\_ClaimReleaseTask](#s4) | The client tries to start a workflow task for a document, claim it and then release it. |

MS-WWSP Scenarios

General precondition

For running the designed scenarios in above table, the following general preconditions of the MS-WWSP test platform must be satisfied first:

* The protocol server is identified by a URL that is known by protocol client.
* MS-WWSP assumes that authentication has been performed by the underlying protocols.

### S01\_StartWorkflow

Description

The client tries to retrieve a set of workflow associations, workflow tasks, and workflows for a document over HTTP as transport and trigger SOAP fault.

Operations

* StartWorkflow
* GetToDosForItem

Prerequisites

N/A.

Cleanup

N/A.

### S02\_GetForItem

Description

The client retrieves workflow's data include task data, template data.

Operations

* GetTemplatesForItem
* StartWorkflow
* GetToDosForItem
* GetWorkflowTaskData
* StartWorkflow

Prerequisites

N/A.

Cleanup

N/A.

### S03\_AlterToDo

Description

The client modifies the values of Fields on a workflow task.

Operations

* AlterToDo
* GetWorkflowTaskData

Prerequisites

N/A.

Cleanup

N/A.

### S04\_ClaimReleaseTask

Description

The client claims or releases a claim on workflow task.

Operations

* GetTemplatesForItem
* StartWorkflow
* GetToDosForItem
* ClaimRleaseTask
* AlterToDo

Prerequisites

N/A.

Cleanup

N/A.

## Test case design

### Traditional test case design

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

12 traditional test cases are designed to cover the four scenarios mentioned in [section 2.5 Test scenarios](#_Test_Scenarios). Details of test cases are specified in [section 2.6.2 Test case description](#_Test_case_description). The scenarios distributions of the test cases are listed in the following table:

|  |  |
| --- | --- |
| Scenario ID | Test case name |
| S01\_StartWorkflow | [MSWWSP\_S01\_TC01\_StartWorkflow\_Success](#s1_tc01) |
| S02\_GetForItem | [MSWWSP\_S02\_TC01\_GetTemplatesForItem\_TemplateData](#s02_tc01) |
| [MSWWSP\_S02\_TC02\_GetToDosForItem\_ToDoData](#s02_tc02) |
| [MSWWSP\_S02\_TC03\_GetWorkflowDataForItem](#s02_tc03) |
| [MSWWSP\_S02\_TC04\_GetWorkflowTaskData\_Success](#s02_tc04) |
| [MSWWSP\_S02\_TC05\_GetWorkflowTaskData\_IgnoreItem](#s02_tc05) |
| S03\_AlterToDo | [MSWWSP\_S03\_TC01\_AlterToDo\_Success](#s03_tc01) |
| [MSWWSP\_S03\_TC02\_AlterToDo\_Fail](#s03_tc02) |
| [MSWWSP\_S03\_TC03\_AlterToDo\_IgnoreItem](#s03_tc03) |
| S04\_ClaimReleaseTask | [MSWWSP\_S04\_TC01\_ClaimReleaseTask\_CorrectURL](#s04_tc01) |
| [MSWWSP\_S04\_TC02\_ClaimReleaseTask\_IgnoreItem](#s04_tc02) |
| [MSWWSP\_S04\_TC03\_ClaimReleaseTask\_AssignedTo](#s04_tc03) |

**Test case scenario distribution**

* The successful test cases are designed to verify the SOAP response messages and the core operations of the protocol. For example, the request message sent to server is actually verified by the server and the response is sent back to the client with correct result.
* Negative Testing is used in the test suite for test cases in S03\_TC02\_AlterToDo\_Fail, the test case aims to verify the soap fault will be returned when the invalid parameter is inputted.

### Test case description

There are 12 traditional test cases designed in this test suite. Common Prerequisites and Common Cleanup for all the test cases are

|  |  |
| --- | --- |
| Common prerequisite | Get current Document List Id, task List Id, workassociation Id |

Common prerequisite

The steps in the following test case use methods and parameters in the adapter interfaces directly.

The following tables describe the traditional test cases.

|  |  |
| --- | --- |
| S01\_StartWorkflow | |
| Test case ID | MSWWSP\_S01\_TC01\_StartWorkflow\_Success |
| Description | This test case is used to verify StartWorkflow operation, starts a new workflow successfully. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT.  2. Call method **GetToDosForItem** to get a set of Workflow Tasks for this document.  **Input parameters:**  • item: upload document URL.  3. Call method **StartWorkflow**  to start a new workflow, generating a workflow from a workflow association.  **Input parameters:**  • item: upload document URL.  • templateId: current workflow association id  • workflowParameters: Load from xml data files.  4. Call method **GetToDosForItem** to get a set of Workflow Tasks for a document.  **Input parameters:**  • item: upload document URL. |
| Cleanup | N/A |

S01\_TC01\_StartWorkflow\_success

|  |  |
| --- | --- |
| S02\_GetForItem | |
| Test case ID | MSWWSP\_S02\_TC01\_GetTemplatesForItem\_TemplateData |
| Description | This test case is used to verify the element TemplateData when GetTemplatesForItem operation successful. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT.  2. Call method **GetTemplatesForItem** to get a set of workflow associations for this existing document.  **Input parameters:**  • item: A document URL. |
| Cleanup | N/A |

S02\_TC01\_GetTemplatesForItem\_TemplateData

|  |  |
| --- | --- |
| S02\_GetForItem | |
| Test case ID | MSWWSP\_S02\_TC02\_GetToDosForItem\_ToDoData |
| Description | This test case is used to verify GetToDosForItem operation to a get a set of workflow tasks for a document. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT.  2. Call method **StartWorkflow** to start a new workflow for this file, generating a workflow from a workflow association.  **Input parameters:**  • item: upload document URL.  • templateId: current workflow association id  • workflowParameters: Load from xml data files.  3. Call method **GetToDosForItem** to get a set of Workflow Tasks for a document.  **Input parameters:**  • item: upload document URL. |
| Cleanup | N/A |

S02\_TC02\_GetToDosForItem\_ToDoData

|  |  |
| --- | --- |
| S02\_GetForItem | |
| Test case ID | MSWWSP\_S02\_TC03\_GetWorkflowDataForItem |
| Description | This test case is used to verify GetWorkflowDataForItem operation, workflow associations, workflow tasks, and workflows should be returned. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT.  2. Call method **StartWorkflow** to start a new workflow for this document, generating a workflow from a workflow association.  **Input parameters:**  • item: upload document URL.  • templateId: current workflow association id  • workflowParameters: Load from xml data files.  3. Call method **GetWorkflowDataForItem** to query a set of workflow associations, workflow tasks, and workflows for a document.  **Input parameters:**  • item: A document URL. |
| Cleanup | N/A |

S02\_TC03\_GetWorkflowDataForItem

|  |  |
| --- | --- |
| S02\_GetForItem | |
| Test case ID | MSWWSP\_S02\_TC04\_GetWorkflowTaskData\_Success |
| Description | This test case is used to verify GetWorkflowTaskData operation, retrieve data about a single workflow task successfully. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT.  2. Call method **StartWorkflow** to start a new workflow for this document, generating a workflow from a workflow association.  **Input parameters:**  • item: upload document URL.  • templateId: current workflow association id  • workflowParameters: Load from xml data files.  3. Call method **GetWorkflowTaskData** to retrieve data about a single workflow task.  **Input parameters:**  • item: A URL to a document on this site.  • taskId: An integer identifying a workflow task.  • listId: A list identifier. |
| Cleanup | N/A |

S02\_TC04\_GetWorkflowTaskData\_Success

|  |  |
| --- | --- |
| S02\_GetForItem | |
| Test case ID | MSWWSP\_S02\_TC05\_GetWorkflowTaskData\_IgnoreItem |
| Description | This test case is used to verify if set the different string as the item value, server reply same when the site of the SOAP request URL contains a list with the specified ListId |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT.  2. Call method **StartWorkflow** to start a new workflow for this document, generating a workflow from a workflow association.  **Input parameters:**  • item: upload document URL.  • templateId: current workflow association id  • workflowParameters: Load from xml data files.  3. Call method **GetWorkflowTaskData** to retrieve data about a single workflow task.  **Input parameters:**  • item: A URL to a document on this site.  • taskId: An integer identifying a workflow task.  • listId: A list identifier.  4. Call method **GetWorkflowTaskData**  to retrieve data about a single workflow task.  **Input parameters:**  • item: not existing document URL.  • taskId: current task id.  • listId: current task list id . |
| Cleanup | N/A |

S02\_TC05\_GetWorkflowTaskData\_IgnoreItem

|  |  |
| --- | --- |
| S03\_AlterToDo | |
| Test case ID | MSWWSP\_S03\_TC01\_AlterToDo\_Success |
| Description | This test case is used to verify AlterToDo operation, modify the values of Fields on a workflow task successful. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT.  2. Call method **StartWorkflow** to start a new workflow for this document, generating a workflow from a workflow association.  **Input parameters:**  • item: upload document URL.  • templateId: current workflow association id  • workflowParameters: Load from xml data files.  3. Call method **AlterToDo** to modify the values of fields on a workflow task.  **Input parameters:**  • item: upload document URL.  • todoId: the id of task item in a task type list which is started in .  • todoListId: list id (GUID format) of the current task list.  • taskData: Contains set of elements representing field names and values to be altered on a workflow task.  4. Call method **GetToDosForItem** to get a set of Workflow Tasks for a document.  **Input parameters:**  • item: upload document URL. |
| Cleanup | N/A |

S03\_TC01\_AlterToDo\_Success

|  |  |
| --- | --- |
| S03\_AlterToDo | |
| Test case ID | MSWWSP\_S03\_TC02\_AlterToDo\_Fail |
| Description | This test case is used to verify AlterToDo operation with the invalid document URL, the operation failed. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT and start workflow for this document.  2. Call method **AlterToDo**  to modify the values of fields on a workflow task.  **Input parameters:**  • item: NULL, this value is no meaning for this operation.  • todoId: Zero, this is invalid value for this operation.  • todoListId: Random GUID value, this is invalid value for this operation.  • taskData: NULL, this means not update content for this operation. |
| Cleanup | N/A |

S03\_TC02\_AlterToDo\_Fail

|  |  |
| --- | --- |
| S03\_AlterToDo | |
| Test case ID | MSWWSP\_S03\_TC03\_AlterToDo\_IgnoreItem |
| Description | This test case is used to verify set the different string as the item value, server reply same if the site of the SOAP request URL contains a list with the specified todoListId. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT and start workflow for this document.  2. Call method **AlterToDo** to modify the values of fields on a workflow task.  **Input parameters:**  • item: upload document URL.  • todoId: the id of task item in a task type list which is started in .  • todoListId: list id (GUID format) of the current task list.  • taskData: Contains set of elements representing field names and values to be altered on a workflow task.  3. Call method **GetToDosForItem** to get a set of Workflow Tasks for a document.  **Input parameters:**  • item: upload document URL.  4. Call method **AlterToDo** to modify the values of fields on a workflow task.  **Input parameters:**  • item: not existing document URL  • todoId: the id of task item in a task type list which is started in .  • todoListId: list id (GUID format) of the current task list.  • taskData: Contains set of elements representing field names and values to be altered on a workflow task.  5. Call method **GetToDosForItem** to get a set of Workflow Tasks for a document.  **Input parameters:**  • item: upload document URL. |
| Cleanup | N/A |

S03\_TC03\_AlterToDo\_IgnoreItem

|  |  |
| --- | --- |
| S04\_ClaimReleaseTask | |
| Test case ID | MSWWSP\_S04\_TC01\_ClaimReleaseTask\_CorrectURL |
| Description | This test case is used to verify ClaimReleaseTask operation when the document URL is correct. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT and start workflow for this document.  2. Call method **ClaimReleaseTask** to Claim or Release a workflow task.  **Input parameters:**  • item: A URL to a document on this site.  • taskId: The list item identifier for a specified workflow task.  • listId: The list identifier for a specified workflow task.  • isClaim: true. |
| Cleanup | N/A |

S04\_TC01\_ClaimReleaseTask\_CorrectURL

|  |  |
| --- | --- |
| S04\_ClaimReleaseTask | |
| Test case ID | MSWWSP\_S04\_TC02\_ClaimReleaseTask\_IgnoreItem |
| Description | This test case is used to verify ClaimReleaseTask operation. The client sets the different strings as the item values, and the server replies the same if the site of the SOAP request URL contains a list with the specified ListId. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT and start workflow for this document.  2. Call method **ClaimReleaseTask** to Claim or Release a workflow task.  **Input parameters:**  • item: current document URL.  • taskId: The list item identifier for a specified workflow task.  • listId: The list identifier for a specified workflow task.  • isClaim: true.  3. Call method **ClaimReleaseTask** sto Claim or Release a workflow task.  **Input parameters:**  • item: not existing document URL.  • taskId: The list item identifier for a specified workflow task.  • listId: The list identifier for a specified workflow task.  • isClaim: false. This will release the claimed task in step3. |
| Cleanup | N/A |

S04\_TC02\_ClaimReleaseTask\_IgnoreItem

|  |  |
| --- | --- |
| S04\_ClaimReleaseTask | |
| Test case ID | MSWWSP\_S04\_TC03\_ClaimReleaseTask\_AssignedTo |
| Description | This test case is used to verify the element AssignedTo when call ClaimReleaseTask operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Upload a file to the SUT and start workflow for this document.  2. Call method **ClaimReleaseTask** to Claim a workflow task.  **Input parameters:**  • item: A URL to a document on this site.  • taskId: The list item identifier for a specified workflow task.  • listId: The list identifier for a specified workflow task.  • isClaim: true.  3. Call method **ClaimReleaseTask** to Release a workflow task.  **Input parameters:**  • item: A URL to a document on this site.  • taskId: The list item identifier for a specified workflow task.  • listId: The list identifier for a specified workflow task.  • isClaim: false. |
| Cleanup | N/A |

S04\_TC03\_ClaimReleaseTask\_AssignedTo