

MS-ASNOTE Test Suite Specification

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

Contents

[1 Configuring the test suite 3](#_Toc387412051)

[1.1 Configuring the test suite client 3](#_Toc387412052)

[1.1.1 Configuring the test suite client manually 3](#_Toc387412053)

[1.1.2 Configuring the test suite client by scripts 3](#_Toc387412054)

[1.2 Configuring the system under test (SUT) 3](#_Toc387412055)

[1.2.1 Configuring the SUT manually 3](#_Toc387412056)

[1.2.2 Configuring the SUT by scripts 3](#_Toc387412057)

[1.3 Configuring the SHOULD/MAY requirements 3](#_Toc387412058)

[2 Test suite design 5](#_Toc387412059)

[2.1 Assumptions, scope and constraints 5](#_Toc387412060)

[2.2 Test suite architecture 5](#_Toc387412061)

[2.3 Technical dependencies and considerations 7](#_Toc387412062)

[2.4 Adapter design 7](#_Toc387412063)

[2.4.1 Adapter overview 7](#_Toc387412064)

[2.4.2 Technical feasibility of adapter approach 7](#_Toc387412065)

[2.4.3 Adapter abstract layer 8](#_Toc387412066)

[2.4.4 Adapter details 8](#_Toc387412067)

[2.5 Test scenarios 9](#_Toc387412068)

[2.5.1 S01\_SyncCommand 9](#_Toc387412069)

[2.5.2 S02\_SearchCommand 9](#_Toc387412070)

[2.5.3 S03\_ItemOperationsCommand 10](#_Toc387412071)

[2.6 Test case design 11](#_Toc387412072)

[2.6.1 Traditional test case design 11](#_Toc387412073)

[2.6.2 Test case description 11](#_Toc387412074)

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

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

Property name="CommonConfigurationFileName" value="ExchangeCommonConfiguration.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 searches through its specific configuration file and uses those properties, if they are defined, before looking for them in the common configuration file (if specified).

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

* Property name="UserName" value="MSASNOTE\_User01"
* Property name="UserPassword" value="Password01!"

### Configuring the test suite client by scripts

To configure the test suite client using scripts, see section 5.2.1 of the [ExchangeEASTestSuiteDeploymentGuide.docx](../ExchangeEASTestSuiteDeploymentGuide.docx).

## Configuring the system under test (SUT)

### Configuring the SUT manually

To manually configure the SUT, see section 5.1.2 of the [ExchangeEASTestSuiteDeploymentGuide.docx](../ExchangeEASTestSuiteDeploymentGuide.docx).

### Configuring the SUT by scripts

To configure the SUT using scripts, see section 5.1.1 of the [ExchangeEASTestSuiteDeploymentGuide.docx](../ExchangeEASTestSuiteDeploymentGuide.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-ASNOTE\_ExchangeServer2010\_SHOULDMAY.deployment.ptfconfig
* MS-ASNOTE\_ExchangeServer2013\_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 Exchange Server 2010 is chosen,user can open **MS-ASNOTE\_ExchangeServer2010\_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 four commands: Sync, Search, FolderSync and ItemOperations.
* This test suite will verify all the XML schema elements embedded inside of the command response.
* 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 requirements from Exchange ActiveSync: Data Types ([MS-ASDTYPE]) and Exchange ActiveSync: WAP Binary XML Algorithm ([MS-ASWBXML]).

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 the Open Specification. The following figure shows the architecture of this test suite.



The architecture of the test suite

The details of the MS-ASNOTE test suite architecture

* SUT hosts the notes data access ActiveSync service which this test suite runs against.
* From third-party user’s point of view, the SUT is the protocol server implementation.
* The following products have been tested with the MS-ASNOTE test suite on the Windows platform.
* Microsoft Exchange Server 2010 Service Pack 3 (SP3)
* Microsoft Exchange Server 2013 Service Pack 1 (SP1)
* The test suite acts as the client to communicate with the SUT and validates the requirements gathered from the MS-ASNOTE Open Specification.
* Test cases use the MS-ASNOTE adapter to call and get the results of the MS-ASNOTE commands.
* MS-ASNOTE adapter is used in the test cases. The test cases call the methods in the interfaces to invoke the MS-ASNOTE protocol adapter’s commands.
* MS-ASNOTE adapter uses ActiveSyncClient to send command request and retrieve command response.
* ActiveSyncClient encodes and decodes Sync command, Search command, FolderSync command and ItemOperations command defined in [MS-ASCMD] by using MS-ASWBXML and communicates with the SUT via MS-ASHTTP.

## Technical dependencies and considerations

Dependencies

* This test suite depends on the HTTP protocol or HTTPS protocol to transmit the messages.
* This test suite depends on the xsd.exe tool in the .NET Framework SDK to generate structures used in the MS-ASNOTE request and response.
* This test suite depends on the Protocol Test Framework (PTF) to derive managed adapter.
* This test suite depends on MS-ASWBXML to encode XMLrequest bodies into WBXML for transmission to an ActiveSync server.
* This test suite depends on MS-ASHTTP to synchronize data that is stored on the server.

Encryption consideration

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

## Adapter design

### Adapter overview

One protocol adapter is used in this test suite.

Protocol adapter

* MS-ASNOTE protocol adapter
* The MS-ASNOTE adapter is a managed adapter, which is derived from the ManagedAdapterBase class in PTF.
* The MS-ASNOTE adapter has the following functionalities
* Choose HTTP or HTTPS for transport;
* Construct requests of the four commands used in MS-ASNOTE;
* Communicate with the SUT by sending requests to the SUT and receiving the corresponding responses from the SUT;
* Parse the response messages and validate the messages according to the XML schema.

### Technical feasibility of adapter approach

Message generation

The MS-ASNOTE adapter gets the parameter values and calls the corresponding commands in ActiveSyncClient. The ActiveSyncClient serializes the parameter values to XML elements, and then encodes request bodies into WBXML for transmission to an ActiveSync server.

Message consumption

The messages received from the SUT will be parsed in the ActiveSyncClient and be passed to the MS-ASNOTE adapter. The messages are then consumed in the MS-ASNOTE adapter to validate the message format and to validate the logic-related requirements in the test cases.

### Adapter abstract layer

Protocol adapter

MS-ASNOTE adapter interface

There are four methods declared in the MS-ASNOTE adapter interface IMS\_ASNOTEAdapter.

Three methods Sync, Search and ItemOperations correspond to the three MS-ASNOTE commands: Sync, Search and ItemOperations. The FolderSync method corresponds to the FolderSync command defined in [MS-ASCMD] which is used to synchronize the collection hierarchy.

The methods are described in the following table.

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
| 1 | Sync | This method is used to synchronize data from the server. |
| 2 | Search | This method is used to search items on the server. |
| 3 | FolderSync | This method is used to synchronize the collection hierarchy. |
| 4 | ItemOperations | This method is used to retrieve data for one or more notes. |

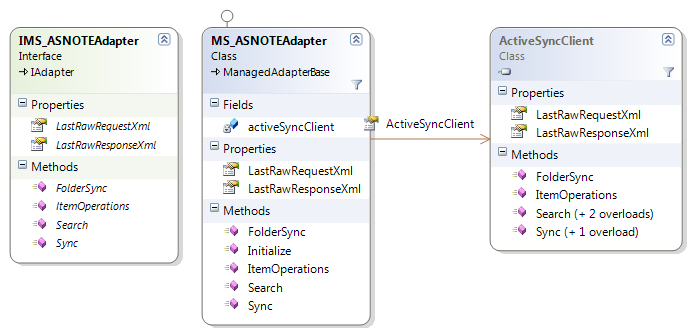
MS-ASNOTE adapter interface methods

### Adapter details

#### Protocol adapter

##### MS-ASNOTE protocol adapter

The following figure illustrates the class diagram of the MS-ASNOTE protocol adapter and the relationship between MS\_ASNOTEAdapter and ActiveSyncClient.



MS-ASNOTE adapter and ActiveSyncClient class diagram

The following outlines details of the class diagram:

Adapter interface

* IMS\_ASNOTEAdapter is the interface of the protocol adapter.
* IMS\_ASNOTEAdapter defines the methods invoked by test cases, including FolderSync, ItemOperations, Search and Sync methods.

Adapter implementation

* MS\_ASNOTEAdapter is the protocol adapter class of the test suite. It is used to implement IMS\_ASNOTEAdapter.
* MS\_ASNOTEAdapter invokes the methods defined in ActiveSyncClient to synchronize or fetch information from the server and search items on the server.
* The Initialize method is used to initialize an instance of ActiveSyncClient.
* The two properties LastRawRequestXml and LastRawResponseXml are used to get the raw request and response xml data.

## Test scenarios

Three scenarios are designed to cover the server-side, testable requirements in the MS-ASNOTE test suite. The details of the scenarios are as follows.

|  |  |
| --- | --- |
| Scenario | Description |
| S01\_SyncCommand | Synchronize Notes class items for a specified user with the existing notes stored on the server. |
| S02\_SearchCommand | Retrieve Notes class items that match the criteria specified by the client. |
| S03\_ItemOperationsCommand | Retrieve data from the server for one or more notes items. |

MS-ASNOTE scenarios

### S01\_SyncCommand

Description

Synchronize Notes class items for a specified user with the existing notes stored on the server.

Commands

* Sync
* FolderSync

Cleanup

Call Sync command to delete the items created in test cases.

### S02\_SearchCommand

Description

Retrieve Notes class items that match the criteria specified by the client.

Commands

* Search
* FolderSync
* Sync

Cleanup

Call Sync command to delete the items created in test cases.

### S03\_ItemOperationsCommand

Description

Retrieve data from the server for one or more notes items.

Commands

* ItemOperations
* FolderSync
* Sync
* Search

Cleanup

Call Sync command to delete the items created in test cases.

## Test case design

### Traditional test case design

Traditional testing is adopted as the test approach for this test suite. The test cases are designed to cover the server-side and testable requirements.

There are 10 traditional test cases designed to cover the three scenarios mentioned in section [2.6 Test scenarios](#_Test_scenarios). Details of the traditional test cases are specified in section [2.7.2 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\_SyncCommand | MSASNOTE\_S01\_TC01\_Sync\_AddNote |
| MSASNOTE\_S01\_TC02\_Sync\_ChangeNote\_WithoutBodyInRequest |
| MSASNOTE\_S01\_TC03\_Sync\_LastModifiedDateIgnored |
| MSASNOTE\_S01\_TC04\_Sync\_SupportedError |
| MSASNOTE\_S01\_TC05\_Sync\_InvalidMessageClass |
| MSASNOTE\_S01\_TC06\_Sync\_AddNote\_WithBodyTypes |
| MSASNOTE\_S01\_TC07\_Sync\_ChangeNote\_Categories |
| S02\_SearchCommand | MSASNOTE\_S02\_TC01\_Search\_GetZeroOrMoreNotes |
| S03\_ItemOperationsCommand | MSASNOTE\_S03\_TC01\_ItemOperations\_GetZeroOrMoreNotes |
| MSASNOTE\_S03\_TC02\_ItemOperations\_SchemaViewFetch |

Test case scenario distribution

### Test case description

The following table describes the common prerequisites and common cleanup for all the test cases.

|  |  |
| --- | --- |
| Common prerequisites | 1. Uses HTTP or HTTPS as the transport; 2. Calls FolderSync command to get ServerId of the Notes folder. |
| Common cleanup | Calls Sync command to delete the items created in test cases. |

Test case common prerequisites and common cleanup

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

The following tables describe the traditional test cases.

|  |  |
| --- | --- |
| S01\_SyncCommand | |
| Test case ID | MSASNOTE\_S01\_TC01\_Sync\_AddNote |
| Description | This test case is designed to test adding a note using the Sync command. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note to the server. 2. The client calls Sync command to synchronize the note item with the server. |
| Cleanup | Common cleanup |

MSASNOTE\_S01\_TC01\_Sync\_AddNote

|  |  |
| --- | --- |
| S01\_SyncCommand | |
| Test case ID | MSASNOTE\_S01\_TC02\_Sync\_ChangeNote\_WithoutBodyInRequest |
| Description | This test case is designed to test changing a note's Subject and MessageClass elements without including the note's body in the Sync command. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note to the server. 2. The client calls Sync command to change the note’s subject, change the message class to IPM.StickyNote.\* and no body element included in request. 3. The client calls Sync command to synchronize the note item with the server and expects to get the updated subject and message class, the body is included in the response too. |
| Cleanup | Common cleanup |

MSASNOTE\_S01\_TC02\_Sync\_ChangeNote\_WithoutBodyInRequest

|  |  |
| --- | --- |
| S01\_SyncCommand | |
| Test case ID | MSASNOTE\_S01\_TC03\_Sync\_LastModifiedDateIgnored |
| Description | This test case is designed to test the server ignores the element LastModifiedDate if includes it in the request. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note which includes the LastModifiedDate element to the server. 2. The client calls Sync command to synchronize the note item with the server and expects that the server ignores the LastModifiedDate in step1. 3. The client calls Sync command to change the LastModifiedDate of the note. 4. The client calls Sync command to get the value of LastModifiedDate and expects the server ignores the value of LastModifiedDate in step3. 5. The client calls Sync command to change the note’s subject and LastModifiedDate elements. 6. The client calls Sync command to synchronize the note item with the server, and expects the subject is updated but the LastModifiedDate is ignored. |
| Cleanup | Common cleanup |

MSASNOTE\_S01\_TC03\_Sync\_LastModifiedDateIgnored

|  |  |
| --- | --- |
| S01\_SyncCommand | |
| Test case ID | MSASNOTE\_S01\_TC04\_Sync\_SupportedError |
| Description | This test case is designed to test when the client includes an airsync:Supported element in a Sync command request, the server returns a status error 4. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note to the server with an airsync:Supported element in the request and expects to get a status error 4 in the response. |
| Cleanup | Common cleanup |

MSASNOTE\_S01\_TC04\_Sync\_SupportedError

|  |  |
| --- | --- |
| S01\_SyncCommand | |
| Test case ID | MSASNOTE\_S01\_TC05\_Sync\_InvalidMessageClass |
| Description | This test case is designed to test when the MessageClass content does not use the standard format in a Sync request, the server responds with a status error 6. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note to the server with an invalid MessageClass element in request and expects to get a status error 6 in the response. |
| Cleanup | Common cleanup |

MSASNOTE\_S01\_TC05\_Sync\_InvalidMessageClass

|  |  |
| --- | --- |
| S01\_SyncCommand | |
| Test case ID | MSASNOTE\_S01\_TC06\_Sync\_AddNote\_WithBodyTypes |
| Description | This test case is designed to test that the type element of the body in note item has 3 different values:1, 2, 3. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note to the server. 2. The client calls Sync command to synchronize the note item with the type in BodyPreference set to 1 and expects to get the body of Type 1. 3. The client calls Sync command to synchronize the note item with the type in BodyPreference set to 2 and expects to get the body of Type 2. 4. The client calls Sync command to synchronize the note item with the type in the BodyPreference set to 3 and expects to get the body of Type 3. |
| Cleanup | Common cleanup |

MSASNOTE\_S01\_TC06\_Sync\_AddNote\_WithBodyTypes

|  |  |
| --- | --- |
| S01\_SyncCommand | |
| Test case ID | MSASNOTE\_S01\_TC07\_Sync\_ChangeNote\_Categories |
| Description | This test case is designed to test changing a note's Categories element and its child elements. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note with two child elements in a Categories element to the server. 2. The client calls Sync command to synchronize the note item with the server and expects to get two child elements in response. 3. The client calls Sync command to change the note with MessageClass elements and one child element of Categories element is missing. 4. The client calls Sync command to synchronize the note item and expects that one child element of Categories element is missing in response. 5. The client calls Sync command to change the note with MessageClass elements and without Categories element. 6. The client calls Sync command to synchronize the note item and expects that the Categories element is missing in response. |
| Cleanup | Common cleanup |

MSASNOTE\_S01\_TC07\_Sync\_ChangeNote\_Categories

|  |  |
| --- | --- |
| S02\_SearchCommand | |
| Test case ID | MSASNOTE\_S02\_TC01\_Search\_GetZeroOrMoreNotes |
| Description | This test case is designed to test when there is zero or more notes that satisfy the search criteria, the server will respond with expected number of notes. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add two notes to the server. 2. The client calls Search command to search notes using the given keyword text and expects to get two notes in response. 3. Call Search command to serach notes using an invalid keyword text. |
| Cleanup | Common cleanup |

MSASNOTE\_S02\_TC01\_Search\_GetZeroOrMoreNotes

|  |  |
| --- | --- |
| S03\_ItemOperationsCommand | |
| Test case ID | MSASNOTE\_S03\_TC01\_ItemOperations\_GetZeroOrMoreNotes |
| Description | This test case is designed to test when there is zero or more notes that satisfy the ItemOperations criteria, the server responds with expected number of notes. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add two notes to the server. 2. The client calls Sync command to synchronize the note items with the server. 3. The client calls ItemOperations command to fetch all the information about notes using the ServerIds returned in step 1 and expects to get two notes in response. 4. The client calls ItemOperations command to fetch all the information about notes using a non-existing ServerIds. |
| Cleanup | Common cleanup |

MSASNOTE\_S03\_TC01\_ItemOperations\_GetZeroOrMoreNotes

|  |  |
| --- | --- |
| S03\_ItemOperationsCommand | |
| Test case ID | MSASNOTE\_S03\_TC02\_ItemOperations\_SchemaViewFetch |
| Description | This test case is designed to test when an itemoperations:Schema element is included in the ItemOperations command request, the elements returned by the server are restricted by the schema. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Sync command to add a note to the server. 2. The client calls Search command to search notes using the given keyword text and expects to get a note and a longId element in response. 3. The client calls ItemOperations command to fetch the elements restricted by the schmea for note using longId. |
| Cleanup | Common cleanup |

MSASNOTE\_S03\_TC02\_ItemOperations\_SchemaViewFetch