Test Specifications for CATS
June, 2012
Version 1.2.3

Document Release Note

CDMI compliant cloud storage vendors:
Project: Storage CoE, HTSC, HiTech ISU

Document Details

Name	Version Number	Description
Test Specifications for CATS	1.2.3	Test specification document for CATS (CDMI Automated Test Suite) solution of Storage CoE, HTSC, HiTech ISU

Revision Details

Action Taken (add/del/change)	Previous page number	New page number	Revision description

Change Register serial numbers covered:
The documents or revised pages are subject to document control.
Please keep them up-to-date using the release notices from the distributor of the document.
These are confidential documents. Unauthorised access or copying is prohibited.

Approved by:
Date:

Authorized by:
Date:

PF2060C

Document Revision List

CDMI compliant cloud storage vendors:
Project: Tata Consultancy Services Ltd.
Document Name: Test Specifications for CATS

Release Notice Reference (for release)

Revision Number	Revision Date	Revision Description	Page Number	Previous Page Number	Action Taken	Addenda/ New Page	Release Notice Reference

About this Document

Purpose

The purpose of this document is to describe each test case in detail.

Intended Audience:

Technical Architects of companies having CDMI compliant Server.

Feedback and Suggestions

For any queries, feedback and suggestions related to the document kindly contact:
storage.coe@tcs.com

Contents

1. **OBJECTIVE** 7
2. **SCENARIOS** 8
3. **SCENARIO SPECIFICATION** 25
4. **REFERENCES** 87

1. Objective

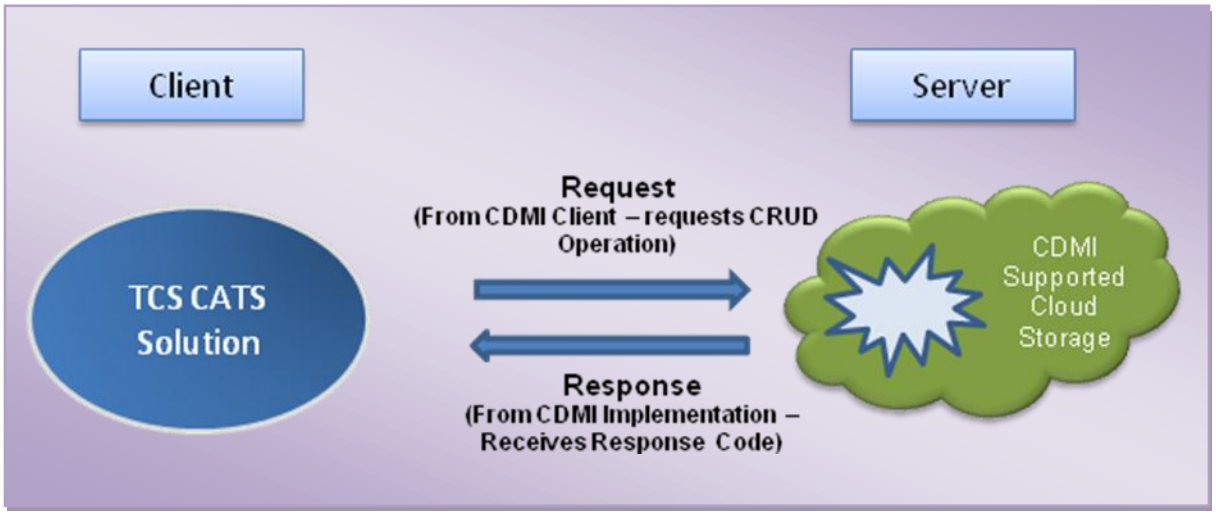
The objective of TCS-CDMI Automated Test Suite (CATS) solution is to validate the degree to which a CDMI storage server implements the Storage Networking Industry Association (SNIA) defined Cloud Data Management Interface (CDMI) standards. The CATS solution has been developed using CDMI specification version 1.0.1.

There are 5 types of objects which can be accessed by unique Uniform Resource Identifier (URI):

- Capability object
- Container object
- Data object
- Queue object
- Domain object

This document has ***limited test cases*** from the overall test suite; a few of the positive and negative scenarios based on Create, Read, Update and Delete (CRUD) operations performed on Capability, Container, Data Object, Queue and domain storage objects have been covered.

The deployment diagram of CATS solution is as follows:



2. Scenarios

Based on the CRUD operations that can be performed on objects, scenarios have been divided as follows

2.1 Capability_Read

The sample case list of Capability_Read scenario is as follows:

Table 1 – Capability_Read

	Case List				
S. No	**Case ID**	**Description**	**Instructions**	**Expected Results**	**Test Result**
1	Capability_Read_4	To read children of capability object up to a specific range.	Get <root URI>/<Capability Object Path/>/cdmi_capabilities/?children:<range>	1.Http Status Code should be 200. 2. Children of System capability object should be listed upto range specified.	
2	Capability_Read_3	To read capability object(system wide capabilities)	Get <root URI>/<Capability Object Path/>	1.Http Status Code should be 200. 2. System wide capabilities should be listed.	

2.2 Container_Create

The sample case list of Container_Create scenario is as follows:

Table 2 – Container_Create

	Case List				
S. No	**Case ID**	**Description**	**Instructions**	**Expected Results**	**Test Result**
3	Container_Create_13	To create a container by copying an existing container.	Put <root URI>/<New Container Name>	1.Http Status Code should be 201. 2.Copy of existing container with the name of new container should be created.	
4	Container_Create_14	To move a container to new URI.	Put <root URI>/<New Container Name>	1.Http Status Code should be 201. 2. Existing container object will be relocated to the URI specified in the PUT.	
5	Container_Create_11	To create a container by using reference field with other fields	Put <root URI>/<New Container Name>/	1. Http Status Code should be 400. 2. Container should not be created.	

2.3 Container_Read :

The sample case list of Container_Read scenario is as follows:

Table 3 – Container Read

S. No	Case ID	Description	Instructions	Expected Results	Test Result
6	Container_Read_3	To read a container with invalid Header Field.	Get<root URI>/ <Container Name>	1. Http Status Code should be 406. 2. Container object should not be listed.	

2.4 Container_Update

The sample case list of Container_Update scenario is as follows:

Table 4 – Container_Update

S. No.	Case ID	Description	Instructions	Expected Results	Test Result
7	Container_Update_12	To Create Snapshot	Put <root URI>/<Container Name>	1.Http Status Code should be 204. 2. Snapshot of Container should be created.	
8	Container_Update_10	To update container with invalid Field name in the URI	Put <root URI>/<Container Name>/?field	1. Http Status Code should be 400. 2. Container object should not be	

				updated.	

2.5 Container_Delete

The sample case list of Container_Delete scenario is as follows:

Table 5– Container_Delete

S.No.	Case ID	Description	Instructions	Expected Results	Test Result
9	Container_Delete_7	To delete the 'cdmi_domains' Container	DELETE<root URI>/cdmi_domains	1. Http Status Code should be 400. 2. Container should not be deleted.	

2.6 Container_PostDataObject

The sample case list of Post Data Object scenario is as follows:

Table 6 – Container_PostDataObject

S.No	Case ID	Description	Instructions	Expected Results	Test Result
10	Container_PostDataobject_5	To Create data object in cdmi_objectid by Post ,without domain URI	Post <root URI>/< cdmi_objectid>/	1. Http Status Code should be 400. 2. Data object should not be created.	
11	Container_PostDataobject_7	To Delete data Object in container, which is not	DELETE <root URI>/<Container name>/<data object	1. Http Status Code should be 404.	

		present.	name>	2. Data object should not be deleted	
12	Container_PostDataobject_19	To create data object in cdmi_objectid by Post	POST<root URI>/< cdmi_objectid>/	1. Http Status Code should be 201. 2..Data Object should be created.	
13	Container_PostDataobject_24	To create data object in container by Post ,without slash in URL.	POST <root URI>/< container name>	1. Http Status Code should be 400. 2..Data Object should not be created.	

14	Container_PostDataobject_17	Read data object, created in container by post	Get <root URI>/< Container Name>/<objectid of data object>	1. Http Status Code should be 200. 2. Data object should be listed with mandatory parameters and their values in response body.	

2.7 Container_PostQueueObject

The sample case list of Post Queue Object scenario is as follows:

Table 7 – Container_PostQueueObject

15	Container_PostQueueobject_12	Create Queue object in container by Post with unauthorized credentials	Post<root URI>/ < Container Name >/	1. Http Status Code should be 401. 2. Queue object should not be created in container.	
16	Container_PostQueueobject_8	To Create Queue object in container by Post with invalid fields.	Post<root URI>/ < Container Name >/	1. Http Status Code should be 400. 2. Queue object should not be created in container.	

17	Container_PostQueueobject_9	To Create Queue object in container by Post with invalid header.	Post<root URI>/ < Container Name >/	1. Http Status Code should be 406. 2. Queue object should not be created in container.	
18	Container_PostQueueobject_18	To Read Queue object using object_id in container by post.	Get<root URI>/<cdmi_objectid>/ <object_id of Queue object>	1. Http Status Code should be 200. 2. Queue object should be listed with mandatory parameters and their values in response body. .	

2.8 DataObject_Create

The sample case list of DataObject_Create scenario is as follows:

Table 8 – DataObject_Create

S No.	Case ID	Description	Instructions	Expected Results	Test Result
19	DataObject_Create_6	To create dataobject when more than one optional field exist in request body.	Put <root URI>/<Container Name>/<Data Object Name>	1. Http Status Code should be 400. 2. Data object should not be created.	

20	DataObject_Create_8	To create data object with invalid value fields.	Put <root URI>/<Container Name>/<Data Object Name>	1. Http Status Code should be 400. 2. Data object should not be created.	

2.9 DataObject_Update

The sample case list of DataObject_Update scenario is as follows:

Table 9 – DataObject_Update

S No.	Case ID	Description	Instructions	Expected Results	Test Result
21	DataObject_Update_8	To Update data object by adding new metadata item in URI while preserving existing metadata	Put <root URI>/<Container Name>/<Data Object Name>?metadata:<prefix>	1. Http Status Code should be 204. 2. Data object should be updated.	
22	DataObject_Update_5	To update data object by valid fields in request message body.	Put <root URI>/<Container Name>/<Data Object Name>	1. Http Status Code should be 204. 2. Data object should be updated .	
23	DataObject_Update_10	To update data object with invalid fields in	Put <root URI>/<Container Name>/<Data Object Name>?	1. Http Status Code should be 400.	

		URI .	<invalid fields>.	2. Data object should not be updated .	

2.10 DataObject_Delete

The sample case list of DataObject_Delete scenario is as follows:

Table 10 – DataObject_Delete

S No.	Case ID	Description	Instructions	Expected Results	Test Result
24	DataObject_Delete_5	To Delete an existing data object	DELETE <root URI>/<Container Name>/<Data Object Name>	1. Http Status Code should be 204. 2. Data object should be deleted.	
25	DataObject_Delete_4	To delete data object that is not present	DELETE <root URI>/<Container Name>/<Data Object Name>	1. Http Status Code should be 404. 2. Data object is not found and not deleted.	

2.11 DataObject_Read

The sample case list of DataObject_Read scenario is as follows:

Table 11 – DataObject_Read

	Case List				
S.No	Case ID	Description	Instructions	Expected Results	Test Results
26	DataObject_Read_4	To read data object that is not present	Get<root URI>/<Container Name>/	1. Http Status Code should be 404. 2. Data object should	

			<data object name>	not be listed.	
27	DataObject _Read_8	To read data object with wrong credentials.	Get<root URI>/<Container Name>/<Data Object Name>	1. Http Status Code should be 401. 2. Data object should not be listed.	

2.12 Queue Object_Create:

The sample case list of Queue_Create scenario is as follows:

Table 12 – Queue Object _Create

S No	Case ID	Description	Instructions	Expected Results	Test Result
28	QueueObject_Create_1	To Create Queue object with invalid Field.	Put <root URI>/Container Name>/<Queue Object Name>	1. Http Status Code should be 400. 2. Queue object should not be created.	
29	QueueObject_Create_4	To Create Queue object	Put <root URI>/Container Name>/<Queue Object Name>	1. Http Status Code should be 201. 2. Queue object should be created.	

2.13 Queue Object _Read

The sample case list of Queue Object _Read scenario is as follows:

Table 13 – Queue Object _Read

	Case ID	Description	Instructions	Expected Results	Test Result
30	QueueObject_Read_2	To Read Queue Object with invalid header field.	Get <root URI>/Container Name>/<Queue Object Name>	1. Http Status Code should be 406. 2. Queue object should not be listed.	

2.14 Queue Object _Update :

The sample case list of Queue Object _Update scenario is as follows:

Table 14 – Queue Object _Update

S No.	Case ID	Description	Instructions	Expected Results	Test Result

31	QueueObject_Update_4	To Update Queue Object when more than one optional field are specified together in request body.	<PUT <root URI>/Container name/<Queue object name>	1.Http Status Code should be 400. 2. Queue object should not be updated.	
32	QueueObject_Update_8	To Update Queue Object with valid fields in request body.	PUT <root URI>/Container name/<Queue object name>	1.Http Status Code should be 204. 2. Queue object should be updated.	
33	QueueObject_Update_9	To Update Queue Object with valid fields in URI.	PUT <root URI>/Container name/<Queue object name>/? <Valid fields>	1.Http Status Code should be 204. 2. Queue object should be updated.	

2.15 Queue Object _Delete:

The sample case list of Queue Object _Delete scenario is as follows:

Table 15 – Queue Object _ Delete

S No.	Case ID	Description	Instructions	Expected Results	Test Result

34	QueueObject_Delete_4	To Delete Queue Object which is not present.	Delete <root URI>/Container name/<Queue object name>	1.Http Status Code should be 404.	
35	QueueObject_Delete_5	To Delete Queue Object.	Delete <root URI>/Container name/< Queue object name >	1.Http Status Code should be 204. 2. Queue object should be deleted.	

2.16 Queue Object _Enqueue

The sample case list of Queue Object _Enqueue scenario is as follows:

Table 16 – Queue Object _Enqueue

	Case ID	Description	Instructions	Expected Results	Test Result
36	QueueObject_Enqueue _5	To enqueue a new value into an existing queue object.	POST <root URI>/Container Name/<Queue Object name>	1. Http status code- 204 should be returned. . 2. New value is enqueued into existing queue object. .	

2.17 Queue Object _Dequeue

The sample case list of Queue Object _Dequeue scenario is as follows:

Table 17 – Queue Object _Dequeue

	Case ID	Description	Instructions	Expected Results	Test Result
37	QueueObject_Dequeue _6	To Dequeue oldest value from an existing queue object.	DELETE <root URI>/Container Name/<Queue Object name>? value	1. Http status code- 204 should be returned. . 2. Oldest value is successfully	

				dequeued from queue object .	

2.18 Domain Object_Create

The sample case list of Domain Object_Create scenario is as follows:

Table 18 – Domain Object_Create

S No.	Case ID	Description	Instructions	Expected Results	Test Result
38	DomainObject _Create_8	To Create a Domain object without slash in URI	Put <root URI>/cdmi_domains/<Domain object name>	1. Http Status Code should be 400. 2. Domain object should not be created.	
39	DomainObject _Create_4	To Create a Domain object	Put <root URI>/cdmi_domains/<Domain name>	1. Http Status Code should be 201. 2. Domain object should be created with mandatory parameters and their values in response body.	
40	DomainObject _Create_9	To create a Domain object by copying existing domain object.	Put <root URI>/cdmi_domains/<Domain name>	1. Http Status Code should be 200. 2. Domain object should be created as a copy of existing domain object.	

2.19 Domain Object_Read

The sample case list of Domain Object_Read scenario is as follows:

Table 19 – Domain Object_Read

S No.	Case ID	Description	Instructions	Expected Results	Test Result

41	Domain Object_Read_1	To Read the Domain Object by using object_id.	GET<root URI>/ cdmi_objectid/<object_id of domain object>	1. Http Status Code should be 200. 2. Domain object should be listed.	
42	Domain Object_Read_3	To Read the Domain Object with invalid header field	GET<root URI>/ cdmi_domains/<Domain name>/	1. Http Status Code should be 406. 2. Domain object should not be listed.	
43	Domain Object_Read_6	To Read the Domain Object	GET<root URI>/ cdmi_domains/<Domain name>/	1. Http Status Code should be 200. 2. Domain object should be listed with mandatory parameters and their values in response body.	
44	Domain Object _Read_7	To Read Domain object when metadata field names are specified in URL	GET<root URI>/ cdmi_domains/Domain name/?metadata : <prefix>	1. Http Status Code should be 200. 2. Domain object metadata should be listed in response body.	
45	Domain Object _Read_8	To Read Domain object with wrong	GET<root URI>/ cdmi_domains/<Domain name>/	1. Http Status Code should be 401.	

		credentials		2. Domain object should not be listed.	

2.20 Domain Object_Update:
The sample case list of Domain Object_Update scenario is as follows:

Table 20 – Domain Object _Update

S No.	Case ID	Description	Instructions	Expected Results	Test Result
46	DomainObject_Update_1	To Update a domain object by object id.	PUT<rootURI>/ <cdmi_objectid>/<object _id of domain object>	1. Http Status Code should be 204. 2.Domain object should be updated.	
47	DomainObject _Update_3	To Update a domain object with invalid fields in URI	PUT<rootURI>/ cdmi_domains/Domain name/?<invalid fields>	1. Http Status Code should be 400. *2*.Domain object should not be updated.	
48	DomainObject _Update_10	To Update a domain object without slash in URI	PUT<rootURI>/ cdmi_domains/<Domain name>	1. Http Status Code should be 400. *2*.Domain object should not be updated.	

2.21 Domain Object Delete:

The sample case list of Domain Object _Delete scenario is as follows:

Table 21 – Domain Object _Delete

S No	Case ID	Description	Instructions	Expected Results	Test Result
49	DomainObject _Delete_1	To Delete an existing domain object by object id.	DELETE<root URI>/ cdmi_domains/<objectid of domain object>	1. Http Status Code should be 204. 2.Domain object should be deleted	
50	DomainObject _Delete_3	To Delete domain object with unauthenticated credentials	DELETE<root URI>/ cdmi_domains/<domain object name>	1. Http Status Code should be 403. 2.Domain object should not be deleted	

3. Scenario Specification

3.1 Capability_Read

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	Capability_01	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Cloud Storage Server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
CapabilityRead_4	To read children of capability object upto specific range. 1. Execute GET request to read capability object using URI - *GET<root URI>/cdmi_capabilities/? <children range>* 2. Check for correct Http status code returned. 3. Verify that required fields for the capability object are returned in the response body.	• Http status code- 200 should be returned. • Children of capability object are returned in response body
CapabilityRead_3	To read capability object(system wide capabilities) 1. Execute GET request to read system wide capability using URI - *GET <root URI>/cdmi_capabilities/* 2. Check for correct Http status code returned. 3. Verify that all mandatory fields for the capability object are returned in the response body.	• Http status code- 200 should be returned. • System wide capabilities should be listed in the response body with all mandatory fields.

3.2 Container_Create

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	Container_01	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Cloud Storage Server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
ContainerCreate_13	To copy an existing container to a new container : 1. Set valid login credentials 2. Test the value of capability “cdmi_create_container” by reading capability object “Container”. 3. Test the value of capability “cdmi_copy_container” by reading capability object “Container”. 4. Execute GET request for container by specifying container name as “TestContainer1” using URI *Get <root URI>/TestContainer1/.* 5. Execute PUT request for container by specifying container name as “TestContainer2” using URI *Put <root URI>/TestContainer1/* 6. Set request message body by specifying ‘copy’ field with the URI of existing container i.e. “TestContainer1” to be copied. 7. Execute PUT request for container by specifying container name as “TestContainer2” using URI *Put <root URI>/TestContainer2/.* 8. Check for correct Http status code returned. 9. Verify that all mandatory fields for the “Testcontainer2” are returned in the response body. 10. Verify if container has been created successfully by executing GET request for “TestContainer2” Using URI- GET<root URI>/TestContainer2/	• User should be able to perform the expected operation. • Capability “cdmi_create_container” should be supported by Cloud storage vendor. • Capability “cdmi_copy_container” should be supported by Cloud storage vendor. • Http status code- 404 should be returned • *TestContainer1 should be created at desired location.* • A copy of ‘TestContainer1’ with the name ‘TestContainer2’ should be created. • Http status code- 201 should be returned. • All mandatory fields for the ‘Testcontainer2’ should be returned in the response body and should be same as that of ‘Testcontainer1’ except for objectId, objectName ,parentID , parentURI. • GET request should be executed successfully to check that container is created.

Case List		
Case Id	**Description**	**Expected Result**
Container_Create_11	To create a new container with 'reference' field along with other fields in request body. 1. Set valid login credentials 2. Test the value of capability "cdmi_create_container" by reading capability object "Container". 3. Test the value of capability "cdmi_create_reference" by reading capability object "Container". 4. Execute PUT request for container1 by specifying container name as "TestContainer1" using URI Put <root URI>/TestContainer1/. 5. Set request message body by specifying 'reference' field as "reference" : "TestContainer1" 6. Set request message body by specifying 'copy' field with the URI of existing container i.e. "TestContainer1" to be copied. 7. Execute PUT request for container2 by specifying container name as "TestContainer2" using URI Put <root URI>/TestContainer2/. 8. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability "cdmi_create_container" should be supported by Cloud storage vendor. • Capability "cdmi_create_reference" should be supported by Cloud storage vendor. • 'TestContainer1' should be created at the desired location. • 'TestContainer2' should not be created. • Http status code- 400 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
Container_Create_14	To move a new container to the URI of existing container. • Set valid login credentials • Test the value of capability "cdmi_create_container" by reading capability object "Container". • Test the value of capability "cdmi_move_container" by reading capability object "Container". • Execute GET request for container by specifying container name as using URI *Get <root URI>/TestContainer1/ TestContainer3/.* • Execute PUT request for container by specifying container name as "TestContainer1" using URI *Put <root URI>/TestContainer1/.* • Execute PUT request for container by specifying another container name as "TestContainer3" with parent directory as previous container name using URI - *Put <root URI>/TestContainer1/TestContainer3/* • Set request message body by specifying 'move' field with the URI of existing container i.e. "TestContainer1/Testcontainer3/" that will be relocated to new URI in Put request. • Execute PUT request for new container "Testcontainer2" using URI - *Put <root URI>/TestContainer1/ TestContainer2/* • Check for correct Http status code returned. • Verify that all mandatory fields for the TestContainer2" are returned in the response body.	• User should be able to perform the expected operation. • Capability "cdmi_create_container" should be supported by Cloud storage vendor. • Capability "cdmi_move_container" should be supported by Cloud storage vendor. • Http status code- 404 should be returned. • 'TestContainer1' should be created at the desired location. • 'TestContainer3' should be created under 'TestContainer1'. • 'TestContainer3' should be moved to the new URI and renamed as 'TestContainer2'. • Http status code- 201 should be returned. • All mandatory fields for the 'Testcontainer2' should be returned in the response body.

3.3 Container_Read

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	Container_02	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
ContainerRead_3	To read a container with invalid header field: • Set valid login credentials • Execute PUT request for container by specifying container name as "TestContainer1" using URI *Put <root URI>/TestContainer1/* • Set invalid header in the GET request • Execute the GET request using URI GET<root URI>/ TestContainer1/ • Check for correct Http status code returned.	• User should be able to perform the expected operation. • 'TestContainer1' should be created at the desired location. • Http Status Code 406 is returned. • Container is not listed.

3.4 Container_Update

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	Container_03	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
ContainerUpdatke_12	To update a container by creating Snapshot : 1. Set valid login credentials 2. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 3. Set ‘snapshots’ field in the PUT request body by specifying the name of the snapshot. 4. Execute the PUT request using URI *Put<root URI>/TestContainer1/* 5. Check for correct Http status code returned.	• User should be able to perform the expected operation. • ‘TestContainer1’ should be created at the desired location. • Container should be updated with snapshot being created at URI : *TestContainer1/cdmi_snapshots /<snapshot name>* • Http Status Code 204 is returned.
ContainerUpdatke_10	To update a container with invalid field names in URI: 6. Set valid login credentials 7. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 8. Set invalid fields in the PUT request URI as well as in request message body. 9. Execute the PUT request using URI *Put<root URI>/TestContainer1/?<invalid field>* 10. Check for correct Http status code returned.	• User should be able to perform the expected operation. • ‘TestContainer1’ should be created at the desired location. • Container is not updated. • Http Status Code 400 is returned.

3.5 Container_Delete

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	Container_04	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
Container_Delete_7	To delete a cdmi_domains container 1. Set valid login credentials 2. Execute DELETE request for container by specifying container name as “cdmi_domains” using URI :- *DELETE <root URI>/cdmi_domains/* 3. Check for correct Http status code returned.	• User should be able to perform the expected operation. • ‘cdmi_domains’ should not be deleted. • Http status code- 400 should be returned.

3.6 Container_Post DataObject

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	Container_05	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
ContainerPostDataobject _7	To delete a data object in container which is not present 1. Set valid login credentials . 2. Execute POST request for container by specifying Container name using URI *POST<root URI>*/TestContainer1/ 3. Execute DELETE request for data object by specifying data object name with parent directory as container name using URI – <DELETE<root URI>/TestContainer1/<data object name> 4. Check for correct Http status code returned.	• User should be able to perform the expected operation. • 'TestContainer1' should be created at the desired location. • Data object is not found for DELETE request. • Http status code- 404 should be returned.
ContainerPostDataobject _5	To create a data object in cdmi_objectid by post without domain URI. 5. Set valid login credentials 6. Execute POST request using URI POST <root URI>/cdmi_objectid/ 7. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Data object should not be created under 'cdmi_objectid' • Http status code- 400 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
ContainerPostDataobject_17	To read a data object created in container by Post : 8. Set valid login credentials 9. Execute PUT request for container by specifying container name as "TestContainer1" using URI *Put <root URI>/TestContainer1/* 10. Execute POST request for data object using URI *POST <root URI>/TestContainer1/* 11. Retrieve objectid of the data object. *12.* Execute the GET request using URI *GET<rootURI>/<ContainerName>/<objectid of data object>* 13. Check for correct Http status code returned.	• User should be able to perform the expected operation. • 'TestContainer1' should be created at the desired location. • Data object with the name of objectid should be created under 'TestContainer1' • Data object should be listed with all the mandatory fields in response body. • Http Status Code 200 should be returned.
ContainerPostDataobject_19	To create a data object in cdmi_objectid by post . 1. Set valid login credentials 2. Execute POST request using URI POST <root URI>/cdmi_objectid/ 3 Check for correct Http status code returned.	• User should be able to perform the expected operation. • Data object with the name of objectid should be created under 'cdmi_objectid'. • Http Status Code 201 is returned.

Case List		
Case Id	**Description**	**Expected Result**
ContainerPostDataobject_24	To create data object in container without slash in URL. 1. Set valid login credentials. 2. Test the value of capability “cdmi_post_dataobject” by reading capability object “Container”. 3. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put<root URI>/TestContainer1/* 4. Execute POST request to create data object using URI.- *Post <root URI>/TestContainer1* 5. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_post_dataobject” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • New data object should not be created under container ‘TestContainer1’. • Http status code- 400 should be returned.

3.7 Container_Post QueueObject

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	Container_06	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
ContainerPost Queueobject_8	To Create Queue object in container by Post with invalid fields. • Set valid login credentials • Test the value of capability “cdmi_post_queue” by reading capability object “Container”. • Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* • Set invalid fields in POST request message body. • Execute POST request to create queue object using following URI with invalid field in Request body.- *Post <root URI>/TestContainer1/* • Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_post_queue” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • New queue object should not be created under container ‘TestContainer1’. • Http status code- 400 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
ContainerPost Queueobject_9	To create a queue object in container by POST with invalid header. 1. Set valid login credentials 2. Test the value of capability “cdmi_create_queue” by reading capability object “Container”. 3. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 4. Set invalid header in the POST request. 5. Execute the POST request to create queue object using URI – *Post <root URI>/TestContainer1/* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_create_queue” should be supported by cloud storage vendor • ‘TestContainer1’ should be created at the desired location. • Queue object is not created. • Http Status Code 406 is returned.
ContainerPost Queueobject_12	To create a queue object in container by post with incorrect authentication. 1. Test the value of capability “cdmi_post_queue” by reading capability object “Container”. 2. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 3. Set invalid credentials in POST request. 4. Execute POST request to create queue object using URI - *Post <root URI>/TestContainer1/* 5. Check for correct Http status code returned.	• Capability “cdmi_post_dataobject” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • New queue object should not be created under container ‘TestContainer1’. • Http status code- 401 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
ContainerPost Queueobject_18	To read a queue object created in container by Post : 1. Set valid login credentials 2. Execute PUT request for container by specifying container name as "TestContainer1" using URI *Put <root URI>/TestContainer1/* 3. Execute POST request to create queue object using URI *POST<root URI>/TestContainer1/* 4. Retrieve objectid of the queue object. *5.* Execute the GET request using URI *GET<rootURI>/<ContainerName>/ <objectid of queue object>* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • 'TestContainer1' should be created at the desired location. • Queue object with the name of objectid should be created under 'TestContainer1' • Successfully get the object id of Queue object. • Queue object should be listed with all the mandatory fields in response body. • Http Status Code 200 should be returned.

3.8 DataObject_Create

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DataObject_01	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials

- IP address of the CDMI complaint server
- Port no
- Company name
- Logging path
- TLS flag

Case List		
Case Id	**Description**	**Expected Result**
DataObjectCreate_6	To create a data object when more than one field (optional) are specified together. 1. Set valid login credentials 2. Test the value of capability "cdmi_create_dataobject" by reading capability object "Container". 3. Execute PUT request for container by specifying container name as "TestContainer1" using URI *Put <root URI>/TestContainer1/* 4. Set more than one fields in request message body for the PUT request. 5. Execute the PUT request for data object by specifying data object name ("do1") and its parent directory as container name using URI – *<PUT <root URI>/TestContainer1/do1* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability "cdmi_create_dataobject" should be supported by cloud storage vendor. • 'TestContainer1' should be created at the desired location. • Data object is not created • Http Status Code 400 is returned .

DataObjectCreate_8	To create a data object with invalid value field. 1. Set valid login credentials. 2. Test the value of capability "cdmi_create_dataobject" by reading system wide capability object. 3. Execute PUT request for container by specifying container name as "TestContainer1" using URI *Put <root URI>/TestContainer1/* 4. Set invalid field in request message body for PUT request. 5. Execute the PUT request to create data object by specifying data name as 'do1' using URI: *PUT<root URI>/ TestContainer1/do1* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability "cdmi_create_dataobject" should be supported by cloud storage vendor. • 'TestContainer1' should be created at the desired location. • Data object is not created Http Status Code 400 is returned.

3.9 DataObject_Update

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DataObject_02	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**

Case List		
Case Id	**Description**	**Expected Result**
DataObject_Update_5	To update valid fields of a data object by passing them in request message body : 1. Set valid login credentials. 2. Execute PUT request for container by specifying container name as “TestContainer1” using URI – *PUT <root URI>/TestContainer1/* 3. Execute PUT request for data object by specifying data object name (“do1”) and its parent directory as container name using URI – *PUT <root URI>/TestContainer1/do1* 4. Set valid fields in request body for PUT request. 5. Execute PUT request to update data object using following URI – *PUT <root URI>/TestContainer1/do1* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • ‘TestContainer1’ should be created at the desired location. • ‘do1’ should be created under container ‘TestContainer1’. • Data object is successfully updated. • Http status code- 204 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
DataObject_Update_8	To update metadata of a data object by adding new metadata item in URI while preserving existing metadata : 1. Set valid login credentials. 2. Execute PUT request for container by specifying container name as “TestContainer1” using URI – *PUT <root URI>/TestContainer1/* 3. Execute PUT request for data object by specifying data object name (“do1”) and its parent directory as container name using URI – PUT <root URI>/TestContainer1/do1 4. Set new metadata fields in request body for PUT request. 5. Execute PUT request to update data object using following URI – <PUT <root URI>/TestContainer1/do1?metadata	• User should be able to perform the expected operation. • ‘TestContainer1’ should be created at the desired location. • ‘do1’ should be created under container ‘TestContainer1’. • Data object is successfully updated. • Http status code- 204 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
DataObject_Update_10	To Update Data Object with invalid fields in URI. 1. Set valid login credentials. 2. Execute PUT request for container by specifying container name as “TestContainer1” using URI – *PUT <root URI>/TestContainer1/* 3. Execute PUT request for data object by specifying data object name (“do1”) and its parent directory as container name using URI – *PUT <root URI>/TestContainer1/do1* 4. Execute PUT request to update data object using following URI – PUT <root URI>/TestContainer1/do1?<invalid fields> 5. Check for correct Http status code returned.	• User should be able to perform the expected operation. • ‘TestContainer1’ should be created at the desired location. • ‘do1’ should be created under container ‘TestContainer1’. • Data object is not updated. • Http status code-400 should be returned.

3.10 DataObject_Delete

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DataObject_03	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
DataObject_delete_5	To delete an existing a data object: 1. Set valid login credentials. 2. Test the value of capability "cdmi_delete_dataobject" by reading capability object "DataObject". 3. Execute PUT request for container by specifying container name as "TestContainer1" using URI – *PUT <root URI>/TestContainer1/* 4. Execute PUT request for data object by specifying data object name ("do1") and its parent directory as container name using URI – *<PUT <root URI>/TestContainer1/do1* 5. Execute DELETE request for data object by specifying data object name with parent directory as container name using URI – *<DELETE<root URI>/TestContainer1/do1* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability "cdmi_delete_dataobject" should be supported by cloud storage vendor. • 'TestContainer1' should be created at the desired location. • 'do1' should be created under container 'TestContainer1' • Http status code- 204 should be returned. • Data object is successfully deleted .

Case List		
Case Id	**Description**	**Expected Result**
DataObject_delete_4	To delete a data object that is not present. 1. Set valid login credentials. 2. Test the value of capability "cdmi_delete_dataobject" by reading capability object "DataObject". 3. Execute PUT request for container by specifying container name as "TestContainer1" using URI – *PUT <root URI>/TestContainer1/* 4. Execute DELETE request for data object by specifying data object name with parent directory as container name using URI – <DELETE<root URI>/TestContainer1/<data object name> 5. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability "cdmi_delete_dataobject" should be supported by cloud storage vendor. • 'TestContainer1' should be created at the desired location. • Data object is not found for DELETE request. • Http status code- 404 should be returned.

3.11 DataObject_Read

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DataObject_04	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server

- Port no
- Company name
- Logging path
- TLS flag

Case List		
Case Id	**Description**	**Expected Result**
DataObjectRead_4	To Read Data Object which is not present. 1. Set valid login credentials. 2. Execute PUT request for container by specifying container name as “TestContainer1” using URI – *PUT <root URI>/TestContainer1/* 3. Execute GET request for data object by specifying data object name with parent directory as container name using URI – <GET<root URI>/TestContainer1/<data object name> 4. Check for correct Http status code returned.	• User should be able to perform the expected operation. • ‘TestContainer1’ should be created at the desired location. • Data object is not found for GET request. • Http status code- 404 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
DataObjectRead_8	To Read Data Object with wrong credentials. 1. Execute PUT request for container by specifying container name as “TestContainer1” using URI – PUT <root URI>/TestContainer1/ 2. Execute PUT request for data object by specifying data object name (“do1”) and its parent directory as container name using URI – <PUT <root URI>/TestContainer1/do1 3. Set invalid credentials in GET request 4. Execute GET request for data object by specifying data object name with parent directory as container name using URI – <GET<root URI>/TestContainer1/do1 5. Check for correct Http status code returned.	• ‘TestContainer1’ should be created at the desired location. • ‘do1’ should be created under ‘TestContainer1’ • Data Object should not be listed. • Http Status Code 401 is returned.

3.12 QueueObject_Create

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	QueueObject_01	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment				

	• 100/1000 Mbps LAN

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

QueueObjectCreate_1	To create a Queue object with invalid field : 1. Set valid login credentials. 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Test the value of capability “cdmi_create_queue” by reading capability object “Container”. 4. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 5. Set invalid field in request message body for PUT request. 6. Execute the PUT request to create queue object by specifying queue name as ‘qo1’ using URI: *PUT<root URI>/ TestContainer1/qo1* 7. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • Capability “cdmi_create_queue” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • Queue object is not created • Http Status Code 400 is returned.

QueueObjectCreate_4	To create Queue Object 1. Set valid login credentials. 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Test the value of capability “cdmi_create_queue” by reading capability object “Container”. 4. Execute PUT request for container by specifying container name as “TestContainer1” using URI Put <root URI>/TestContainer1/ 5. Execute PUT request for queue object by specifying queue object name as “qo1” using URI Put<root URI>/TestContainer1/qo1 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • Capability “cdmi_create_queue” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • ‘qo1’ should be created in ‘TestContainer1’. • Http Status Code 201 is returned.

3.13 QueueObject_Read

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	QueueObject_02	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation

Update the following contents in Configuration file:

- Server name
- Set Valid login credentials
- Set Valid but unauthorized login credentials
- IP address of the CDMI complaint server
- Port no
- Company name
- Logging path
- TLS flag

Case List		
Case Id	**Description**	**Expected Result**
QueueObject_ Read_2	To Read Queue Object with Invalid header field. 1 Set valid login credentials 2 Test the value of capability “cdmi_ queues” by reading system wide capability. 3 Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 4 Execute PUT request for queue object by specifying queue object name as “qo1” using URI *Put <root URI>/TestContainer1/qo1* 5 Set invalid header in the GET request 6 Execute the GET request using URI GET<root URI>/ TestContainer1/qo1 7 Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • Queue object ‘qo1’ should be created under ‘TestContainer1’ • Http Status Code 406 is returned. • Queue object is not listed in response body.

3.14 QueueObject_Update

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	QueueObject_03	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**

Case List		
Case Id	**Description**	**Expected Result**
QueueObject_ Update_4	To update a queue object when more than one field (optional) are specified together. 1. Set valid login credentials 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 4. Execute PUT request for Queue object by specifying queue name as “qo1” using URI *Put <root URI>/TestContainer1/qo1* 5. Set more than one fields in request message body for the PUT request. 6. Execute the PUT request to update queue object using URI – *PUT <root URI>/TestContainer1/qo1* 7. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • ‘qo1’ should be created under ‘TestContainer1’. • Queue object is not updated • Http Status Code 400 is returned.

Case List		
Case Id	**Description**	**Expected Result**
QueueObject_ Update_8	To Update Queue Object with valid fields in request message body. 1. Set valid login credentials 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 4. Execute PUT request for Queue object by specifying queue name as “qo1” using URI *Put <root URI>/TestContainer1/qo1* 5. Set valid fields in request message body for the PUT request. 6. Execute the PUT request to update queue object using URI – *PUT <root URI>/TestContainer1/qo1* 7. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • ‘qo1’ should be created under ‘TestContainer1’. • Queue object is updated Http Status Code 204 is returned.

Case List		
Case Id	**Description**	**Expected Result**
QueueObject_ Update_9	To Update Queue Object with valid fields in URI. 1. Set valid login credentials 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Execute PUT request for container by specifying container name as “TestContainer1” using URI *Put <root URI>/TestContainer1/* 4. Execute PUT request for Queue object by specifying queue name as “qo1” using URI *Put <root URI>/TestContainer1/qo1* 5. Set valid fields in URI for the PUT request. 6. Execute the PUT request to update queue object using URI – PUT <root URI>/TestContainer1/qo1?<valid fields> 7. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • ‘qo1’ should be created under ‘TestContainer1’. • Queue object is updated Http Status Code 204 is returned.

3.15 QueueObject_Delete

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	QueueObject_04	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
QueueObject_ Delete_4	To delete a queue object which is not present. 1. Set valid login credentials 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Test the value of capability “cdmi_delete_queue” by reading capability object “Queue”. 4. Execute PUT request for container by specifying container name as “TestContainer1” *Put <root URI>/Testcontainer1/* 5. Execute DELETE request for queue object using URI : *DELETE <root URI>/TestContainer1/qo1* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • Capability “cdmi_delete_queue” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • Queue object is not found for Delete request. • Http status code- 404 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
QueueObject_ Delete_5	To delete an existing queue object 1. Set valid login credentials 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Test the value of capability “cdmi_delete_queue” by reading capability object “Queue”. 4. Execute PUT request for container by specifying container name as “TestContainer1” *Put <root URI>/Testcontainer1/* 5. Execute PUT request for queue object by specifying queue object name as “qo1” *Put <root URI>/Testcontainer1/qo1* 6. Execute DELETE request for queue object using URI : *DELETE <root URI>/TestContainer1/qo1* 7. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • Capability “cdmi_delete_queue” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • ‘qo1’ should be created in’TestContainer1’. • Queue object is successfully deleted. • Http status code- 204 should be returned.

3.16 QueueObject_Enqueue

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	QueueObject_05	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
QueueObject_enqueue_5	To enqueue a new value into an existing queue object . 1. Set valid login credentials. 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Execute PUT request for container by specifying container name as “TestContainer1” using URI – *PUT <root URI>/TestContainer1/* 4. Execute PUT request for data object by specifying queue object name (“qo1”) and its parent directory as container name using URI – *PUT<root URI>/TestContainer1/qo1.* 5. Set value in request body for POST request for Enqueue. 6. Execute POST request using URI – *POST <root URI>/TestContainer1/qo1* 7. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • ‘qo1’ should be created under container ‘TestContainer1’. • New values are enqueued into existing queue object. • Http status code- 204 should be returned. .

3.17 QueueObject_Dequeue

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	QueueObject_06	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
Queue_dequeue_6	To Dequeue oldest value from an existing queue object. 1. Set valid login credentials. 2. Test the value of capability “cdmi_ queues” by reading system wide capability. 3. Execute PUT request for container by specifying container name as “TestContainer1” using URI – *PUT <root URI>/TestContainer1/* 4. Execute PUT request for data object by specifying queue object name (“qo1”) and its parent directory as container name using URI – *PUT <root URI>/TestContainer1/qo1.* 5. Execute DELETE request to dequeue using URI – *DELETE <root URI>/TestContainer1/qo1 ? value* 6. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_queues” should be supported by cloud storage vendor. • ‘TestContainer1’ should be created at the desired location. • ‘qo1’ should be created under container ‘TestContainer1’. • Oldest value is successfully dequeued from queue object . • Http status code- 204 should be returned. .

3.18 Domain object_Create

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DomainObject_01	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Create_9	To copy an existing domain object to new domain object. 1. Set valid login credentials 2. Test the value of capability “cdmi_ domains” by reading system wide capability. 3. Test the value of capability “cdmi_ create_domain” by reading capability object “Domain” 4. Execute PUT request for domain object by specifying domain name as “TestDomain1” using URI 5. *Put <root URI>/ cdmi_domains/TestDomain1/* 6. Set request message body by specifying ‘copy’ field with the URI of existing domain object i.e. “TestDomain1/”. 7. Execute the PUT request for new domain object by specifying domain object name (“TestDomain2”) using URI – *Put <root URI>/ cdmi_domains/TestDomain2/* 8. Check for correct Http status code returned. 9. Verify that all mandatory fields for the “TestDomain2” are returned in the response body. 10. Execute the GET request by specifying domain object name (“TestDomain2”) using URI – *Put <root URI>/ cdmi_domains/TestDomain2/*	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • Capability “cdmi_create_domain” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • Request body should be successfully set for ‘copy’ field. • ‘TestDomain2’ should be created as a copy of ‘TestDomain1’. • Http Status Code 201 is returned. • All mandatory fields for ‘TestDomain2’ should be returned in response body. • Get request should be executed successfully to verify that ‘TestDomain2’ has been created.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Create_4	To create a domain object: 11. Set valid login credentials 12. Test the value of capability “cdmi_ domains” by reading system wide capability. 13. Test the value of capability “cdmi_create_domain” by reading capability object “Domain”. 14. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1 /* 15. Check for correct Http status code returned. 16. Verify that all mandatory fields for the “TestDomain1” are returned in the response body. 17. Verify if domain object has been created successfully by executing GET request using URI- *GET<root URI>/ cdmi_domains/TestDomain1/*	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • Capability “cdmi_create_domain” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • Http status code- 201 should be returned. • All mandatory fields for the ‘TestDomain1’ should be returned in the response body. • GET request should be executed successfully to check that domain object is created.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Create_8	To create a domain object without slash in URI. 1. Set valid login credentials 2. Test the value of capability “cdmi_ domains” by reading system wide capability. 3. Test the value of capability “cdmi_create_domain” by reading capability object “Domain”. 4. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1* 5. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • Capability “cdmi_create_domain” should be supported by cloud storage vendor. • ‘TestDomain1’ should not be created. • Http status code- 400 should be returned.

3.19 Domain object_Read

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DomainObject_02	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Read_6	To Read a domain object: 6. Set valid login credentials 7. Test the value of capability “cdmi_ domains” by reading system wide capability. 8. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1 /* 9. Execute GET request for domain object using URI *:* *GET<root URI>/ cdmi_domains/TestDomain1/* 10. Check for correct Http status code returned. 11. Verify that all mandatory fields for the “TestDomain1” are returned in the response body.	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • GET request should be executed successfully to read domain object. • Http status code- 200 should be returned. • All mandatory fields for the ‘TestDomain1’ should be returned in the response body.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Read_7	To Read particular fields of metadata of domain object: 1. Set valid login credentials. 2. Test the value of capability “cdmi_ domains” by reading system wide capability. 3. Test the value of capability “cdmi_read_metadata” by reading capability object “Domain”. *4.* Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI : *Put <root URI>/cdmi_domains/TestDomain1 /* 5. Fetch metadata fields to be read, from data file. 6. Execute GET request for domain object using URI : GET<root URI>/ cdmi_domains/TestDomain1/?met adata : <prefix> 7. Check for correct Http status code returned. 8. Verify that required metadata field for the “TestDomain1” is returned in the response body.	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • Capability “cdmi_ read_metadata” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • Metadata field should be successfully fetched from data file and set for GET request URI. • GET request should be executed successfully to read metadata of domain object. • Http status code- 200 should be returned. • Metadata fields for the ‘TestDomain1’ should be returned in the response body.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Read_1	To Read domain object by object_id: 9. Set valid login credentials 10. Test the value of capability “cdmi_ domains” by reading system wide capability. 11. Execute PUT request for domain object using URI. *PUT <root URI>/cdmi_domains/TestDomain1 /* 12. Retrieve object id of the domain object. 13. Test the value of capability “cdmi_object_access_by_ID” by reading system wide capability . *14.* Execute the GET request using URI *GET<root URI>/cdmi_objectid/<objectid of domain object>* 15. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • Domain object_id is successfully retrieved. • Capability “cdmi_object_access_by_ID” should be supported by cloud storage vendor. • Domain object is listed successfully. • .Http Status Code 200 is returned.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Read_3	To Read domain object with invalid header field. 1. Set valid login credentials 2. Test the value of capability “cdmi_ domains” by reading system wide capability. 3. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1 /* 4. Set invalid header in the GET request. 5. Execute GET request for domain object using URI *: GET<root URI>/ cdmi_domains/TestDomain1/* 6. Check for correct Http status code returned. .	• Capability “cdmi_domains” should be supported by cloud storage vendor. • User should be able to perform the expected operation. • ‘TestDomain1’ should be created at the desired location. • GET request should not be executed. • Http status code- 406 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Read_8	To Read domain object with wrong credentials: 16. Test the value of capability “cdmi_ domains” by reading system wide capability. 17. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1 /* 18. Set wrong credentials in the GET Request. 3 .Execute GET request for domain object using URI *: GET<root URI>/ cdmi_domains/TestDomain1/* 7. Check for correct Http status code returned.	• Capability “cdmi_domains” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • GET request should not be executed. • Http status code- 401 should be returned.

3.20 Domain object_Update

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DomainObject _03	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_pl an_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file:

- Server name
- Set Valid login credentials
- Set Valid but unauthorized login credentials
- IP address of the CDMI complaint server
- Port no
- Company name
- Logging path
- TLS flag

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Update_1	To update Domain Object using object_id: 1. Set valid login credentials 2. Test the value of capability “cdmi_ domains” by reading system wide capability. 3. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1 /* 4. Retrieve object id of the domain object. 5. Test the value of capability “cdmi_object_access_by_ID” by reading system wide capability 6. Set valid field in request body for update. 7. Execute PUT request for domain object using URI *: Pur<root URI>/ cdmi_objectid/<objectid of domain>* 8. Check for correct Http status code returned. a. .	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • Domain object id is successfully retrieved. • Capability “cdmi_object_access_by_ID” should be supported by Cloud vendor • Domain object is updated successfully. • .Http Status Code 204 is returned.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Update_3	To update Domain Object with invalid fields in URI: 1. Set valid login credentials 2. Test the value of capability “cdmi_ domains” by reading system wide capability. 3. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put<root URI>/cdmi_domains/TestDomain1 /* 4. Set invalid field in URI for update. 4. Execute PUT request for domain object using URI *: Put<root URI>/ cdmi_domains/TestDomain1/?<Inv alid fields>* 5. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location • Domain object is not updated. • Http Status Code 400 is returned

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Update_10	To update Domain Object without slash in URI: 1. Set valid login credentials 2. Test the value of capability “cdmi_ domains” by reading system wide capability. 3. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1 /* 4. Execute PUT request for domain object using URI *: Put<root URI>/ cdmi_domains/TestDomain1* 5. Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location • Domain object is not updated. • Http Status Code 400 is returned

3.21 Domain object_Delete

This test scenario includes those Http requests which have CDMI content type.

Scenario ID	DomainObject_04	**Version No.**	1.2.3	**Test Plan Ref.**	Compliance_test_plan_ver1.0.2
Environment	Hardware – • Intel Core 2 Duo processor 2.88 GHz • 1 GB RAM Software – • JUnit • JDK 1.5 • CDMI Compliant Storage server Networking Environment • 100/1000 Mbps LAN				

Preparation
Update the following contents in Configuration file: • Server name • Set Valid login credentials • Set Valid but unauthorized login credentials • IP address of the CDMI complaint server • Port no • Company name • Logging path • TLS flag

Case List		
Case Id	**Description**	**Expected Result**

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Delete_3	To delete a domain object with unauthenticated credentials. 6. Test the value of capability “cdmi_ domains” by reading system wide capability. 7. Test the value of capability “cdmi_delete_domain” by reading capability object “Domain”. 8. Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1 /* 9. Set unauthenticated credentials in DELETE request. 10. Execute DELETE request for domain object using URI : *DELETE <root URI>/ cdmi_domains/TestDomain1/* *11.* Check for correct Http status code returned	• Capability “cdmi_domains” should be supported by cloud storage vendor. • Capability “cdmi_delete_domain” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • DELETE request should not be executed • Http status code-403 should be returned.

Case List		
Case Id	**Description**	**Expected Result**
Domainobject_ Delete_1	To delete a domain object by objectid: • Set valid login credentials. • Test the value of capability “cdmi_ domains” by reading system wide capability. • Test the value of capability “cdmi_delete_domain” by reading capability object “Domain”. • Execute PUT request for domain object by specifying domain name as “TestDomain1” with parent directory as “cdmi_domains” using URI *Put <root URI>/cdmi_domains/TestDomain1/* • Retrieve objectid of the domain object. • Execute DELETE request for domain object using URI : *DELETE<root URI>/ cdmi_objectid/<objectid of domain>/* • Check for correct Http status code returned.	• User should be able to perform the expected operation. • Capability “cdmi_domains” should be supported by cloud storage vendor. • Capability “cdmi_delete_domain” should be supported by cloud storage vendor. • ‘TestDomain1’ should be created at the desired location. • Objectid should be successfully retrieved. • DELETE request should be executed successfully to delete domain object. • Http status code- 204 should be returned.

References

[1] CDMI Specification – http://cdmi.sniacloud.com/

[2] Compliance Test Plan, TCS, v1.0.1