RESTful Network API for Unified Cloud Disk

Candidate Version 1.0 – 16 Dec 2014

Open Mobile Alliance

OMA-TS-REST_NetAPI_UCD-V1_0-20141216-C

Contents

Figures

Tables

1. Scope

This specification defines RESTful Network API for Unified Cloud Disk using HTTP protocol bindings.

2. References

2.1 Normative References

[Autho4API_10] "Authorization Framework for Network APIs", Open Mobile Alliance™, OMA-ER-Autho4API-V1_0, URL: http://www.openmobilealliance.org/

[IETF_ACR_draft] Include if the use of ACR is supported, otherwise delete this reference. "The acr URI for anonymous users", S.Jakobsson, K.Smith, January 2010, URL: http://tools.ietf.org/html/draft-uri-acr-extension-00

[REST_NetAPI_Common] "Common definitions for RESTful Network APIs", Open Mobile Alliance™, OMA-TS-REST_NetAPI_Common-V1_0, URL: http://www.openmobilealliance.org/

[REST_SUP_UCD] "XML schema for the RESTful Network API for Unified Cloud Disk", Open Mobile Alliance™, OMA-SUP-XSD_rest_netapi_ucd-V1_0, URL: http://www.openmobilealliance.org/

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt

[RFC2616] "Hypertext Transfer Protocol -- HTTP/1.1", R. Fielding et. al, January 1999, URL:http://www.ietf.org/rfc/rfc2616.txt

[RFC3530] "Network File System (NFS) version 4 Protocol", S. Shepler, April 2003, URL:http://www.ietf.org/rfc/rfc3530.txt

[RFC3966] "The tel URI for Telephone Numbers", H.Schulzrinne, December 2004, URL: http://www.ietf.org/rfc/rfc3966.txt

[RFC3986] "Uniform Resource Identifier (URI): Generic Syntax", R. Fielding et. al, January 2005, URL:http://www.ietf.org/rfc/rfc3986.txt

[RFC4627] "The application/json Media Type for JavaScript Object Notation (JSON)", D. Crockford, July 2006, URL: http://www.ietf.org/rfc/rfc4627.txt

[SCRRULES] "SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/

[W3C_URLENC] HTML 4.01 Specification, Section 17.13.4 Form content types, The World Wide Web Consortium, URL: http://www.w3.org/TR/html401/interact/forms.html#h-17.13.4.1 [only needed if application/x-www-form-urlencoding (Appendix C) is supported]

[XMLSchema1] W3C Recommendation, XML Schema Part 1: Structures Second Edition, URL: http://www.w3.org/TR/xmlschema-1/

[XMLSchema2] W3C Recommendation, XML Schema Part 2: Datatypes Second Edition, URL: http://www.w3.org/TR/xmlschema-2/

2.2 Informative References

[OMADICT] "Dictionary for OMA Specifications", Version 2.8, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_8, URL:http://www.openmobilealliance.org/

[REST_WP] "Guidelines for RESTful Network APIs", Open Mobile Alliance™, OMA-WP-Guidelines_for_RESTful_Network_APIs, URL:http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

For the purpose of this TS, all definitions from the OMA Dictionary apply [OMADICT]. If the use of Notification Channel and/or Light-weight Resources are supported, include also the definitions below, otherwise delete those that are not applicable.

Heavy-weight Resource	A resource which is identified by a resource URL which is then used by HTTP methods to operate on the entire data structure representing the resource. Include this definition if Light-weight Resources are supported, otherwise delete it.
Light-weight Resource	A subordinate resource of a Heavy-weight Resource which is identified by its own resource URL which is then used by HTTP methods to operate on a part of the data structure representing the Heavy-weight Resource. The Light-weight Resource URL can be seen as an extension of the Heavy-weight Resource URL. There could be several levels of Light-weight Resources below the ancestor Heavy-weight Resource, depending on the data structure. Include this definition if Light-weight Resources are supported, otherwise delete it.

3.3 Abbreviations

ACE	Access Control Entries
ACL	Access Control List
ACR	Anonymous Customer Reference
API	Application Programming Interface
HTTP	HyperText Transfer Protocol
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
NFS	Network File System
OMA	Open Mobile Alliance
REST	REpresentational State Transfer
SCR	Static Conformance Requirements
SIP	Session Initiation Protocol
TS	Technical Specification
UCD	Unified Cloud Disk
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WP	White Paper
XML	eXtensible Markup Language
XSD	XML Schema Definition

4. Introduction

The Technical Specification of the RESTful Network API for Unified Cloud Disk contains HTTP protocol bindings for Unified Cloud Disk, using the REST architectural style. The specification provides resource definitions, the HTTP verbs applicable for each of these resources, and the element data structures, as well as support material including flow diagrams and examples using the various supported message body formats (i.e. XML, JSON, and application/x-www-form-urlencoded).

4.1 Version 1.0

Version 1.0 of this specification supports the following operations:

- Folder operations including list folder information, create folder, delete folder, retrieve or update individual folder information
- File operations including download file, upload total file, update file in range with the start byte and the end byte, delete file, retrieve or update individual file information
- Recyclebin operations including list recycle bin, revoke or clean the recycle bin
- Common operations including search file or folder, list all file sharing

The following new functionality has been introduced:

- Support for scope values used with authorization framework defined in [Autho4API_10]
- Support for Anonymous Customer Reference (ACR) as an end user identifier
- Support for “acr:Authorization” as a reserved keyword in an ACR

5. Unified Cloud Disk API definition

This section is organized to support a comprehensive understanding of the Unified Cloud Disk API design. It specifies the definition of all resources, definition of all data structures, and definitions of all operations permitted on the specified resources.

Common data types, naming conventions, fault definitions and namespaces are defined in [REST_NetAPI_Common].

The remainder of this document is structured as follows:

Section 5 starts with a diagram representing the resources hierarchy followed by a table listing all the resources (and their URL) used by this API, along with the data structure and the supported HTTP verbs (section 5.1). What follows are the data structures (section 5.2). A sample of typical use cases is included in section 5.3, described as high level flow diagrams.

Section 6 contains detailed specification for each of the resources. Each such subsection defines the resource, the request URL variables that are common for all HTTP methods, and the supported HTTP verbs. For each supported HTTP verb, a description of the functionality is provided, along with an example of a request and an example of a response. For each unsupported HTTP verb, the returned HTTP error status is specified, as well as what should be returned in the Allow header.

All examples in section 6 use XML as the format for the message body. Application/x-www-form-urlencoded examples are provided in Appendix C, while JSON examples are provided in Appendix D.

Section 7 contains fault definition details such as Service Exceptions and Policy Exceptions.

Appendix B provides the Static Conformance Requirements (SCR).

Appendix F provides a list of all Light-weight Resources, where applicable.

Appendix G defines authorization aspects to control access to the resources defined in this specification.

Note: Throughout this document client and application can be used interchangeably.

5.1 Resources Summary

This section summarizes all the resources used by the RESTful Network API for Unified Cloud Disk.

The "apiVersion" URL variable SHALL have the value “v1" to indicate that the API corresponds to this version of the specification. See [REST_NetAPI_Common] which specifies the semantics of this variable.

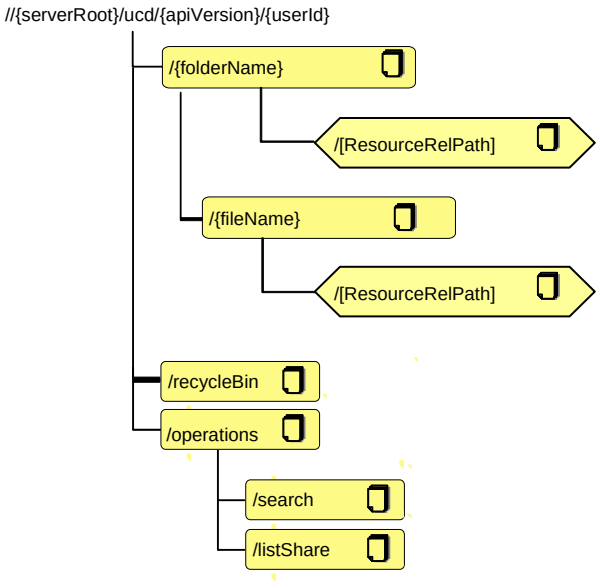

Figure 1 UCD-2 Resource structure defined by this specification

The following tables give a detailed overview of the resources defined in this specification, the data type of their representation and the allowed HTTP methods.

Purpose: To allow client to manage folder or file

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
A folder	/{folderName} Note: folderName is absolute value including path	Folder (used for GETresponse, PUTresponse) common:ResourceReference (optional alternative for PUT response) DeleteMode (used for DELETE request)	List folder information	Create folder	No	Delete folder to recycle bin or permanently
Individual folder attributes	/{folderName}/[ResourceRelPath]	The data structure corresponds to an element within the Folder structure pointed out by the resource URL. (used for PUT/GET)	Retrieve individual folder attributes	Update individual folder ACL information	No	No
A file	/{folderName}/{fileName} Note: folderName is absolute value including path	File (used for PUTresponse) common:ResourceReference (optional alternative for PUT response) DeleteMode (used for DELETE request)	Download file	Upload file Update file in range with the start byte and the end byte	No	Delete file to recycle bin or permanently

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Individual file attributes	/{folderName}/{fileName}/[ResourceRelPath]	The data structure corresponds to an element within the File structure pointed out by the resource URL. (used for PUT/GET)	Retrieve individual file attributes	Update individual file attributes including fileType, metadata, ACL information, sharing information	No	No

Purpose: To allow client to manage Recycle bin

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Recycle bin	/recycleBin	RecycleBin (used for GET response, PUT request)	List recycle bin	Revoke or delete the recycle bin items	No	No

Purpose: To allow client to manage public operations for folder or file

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Search for file or folder	/operations/search	SearchCriteria (used for POST request) SearchResult (used for POST response)	No	No	Search file or folder	No
List of shared files	/operations/listShare	ShareList (used for POST response)	No	No	List all file sharing	No

5.2 Data Types

5.2.1 XML Namespaces

The XML namespace for the Unified Cloud Disk data types is:

urn:oma:xml:rest:ucd:1

The 'xsd' namespace prefix is used in the present document to refer to the XML Schema data types defined in XML Schema [XMLSchema1, XMLSchema2]. The 'common' namespace prefix is used in the present document to refer to the data types defined in [REST_NetAPI_Common]. The use of namespace prefixes such as 'xsd' is not semantically significant.

The XML schema for the data structures defined in the section below is given in [REST_SUP_UCD].

5.2.2 Structures

The subsections of this section define the data structures used in the UCD API.

Some of the structures can be instantiated as so-called root elements, i.e. they define the type of a representation of a so-called Heavy-weight Resource.

The column [ResourceRelPath] in the tables below, if used, includes relative resource paths for Light-weight Resource URLs that are used to access individual elements in the data structure (so-called Light-weight Resources). A string from this column needs to be appended to the corresponding Heavy-weight Resource URL in order to create Light-weight Resource URL for that particular element in the data structure. “Not applicable” means that individual access to that element is not supported. The root element and data type of the resource associated with the [ResourceRelPath] are defined by the Element and Type columns in the row that defines the [ResourceRelPath].

For structures that contain elements which describe a user identifier, the statements in section 6 regarding 'tel', 'sip' and 'acr' URI schemes apply.

5.2.2.1 Type: File

Individual file

Element	Type	Optional	[ResourceRelPath]	Description
fileAttributes	FileAttributes	Yes	fileAttributes	List of attributes associated with the file
resourceURL	xsd:anyURI	Yes	Not applicable	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named file of type File is allowed in response bodies.

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.2 Type: FileAttributes

File attributes.

Element	Type	Optional	[ResourceRelPath]	Description
fileType	xsd:string	Yes	fileAttributes/fileType	The file type, (i.e., jpg, doc, xls, zip).
size	xsd:unsignedInt	Yes	Not applicable	The file size. Indicates the size of the stored content in bytes
createTime	xsd:dateTimeStamp	Yes	Not applicable	Date and Time at which the file was created.
modifyTime	xsd:dateTimeStamp	Yes	Not applicable	Date and Time at which the file was modified.
accessTime	xsd:dateTimeStamp	Yes	Not applicable	Date and Time at which the file was accessed.
owner	xsd:string	Yes	Not applicable	The owner of the file, e.g. userId
metadataList	MetadataList	Yes	fileAttributes/metadata	The user self define metadata, e.g. department, project, group, publisher, editor.
accessControlList	AccessControlList	Yes	fileAttributes/acl	The access control list information.
hash	HashInformation	Yes	Not applicable	The hash information of the file
share	Share	Yes	fileAttributes/share	File Sharing information.
revisionList	ReferenceList	Yes	Not applicable	The file revisions

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.3 Type: MetadataList

The list of metadata information.

Element	Type	Optional	Description
metadata	Metadata [0..unbounded]	Yes	A list of metadata.

5.2.2.4 Type: Metadata

The user self define metadata.

Element	Type	Optional	Description
name	xsd:string	No	Metadata name.
value	xsd:string	Yes	Metadata value.

5.2.2.5 Type: AccessControlList

Access control comprises the mechanisms by which various types of access to objects are authorized and permitted or denied. UCD uses the well-known mechanism of an Access Control List (ACL) as defined in the NFSv4 standard [RFC 3530]. ACLs are lists of permissions-granting or permissions-denying entries called access control entries (ACEs).

Element	Type	Optional	Description
accessControlEntry	AccessControlEntry [0..unbounded]	Yes	A list of resource references

5.2.2.6 Type: AccessControlEntry

The access control entry information.

Element	Type	Optional	Description
acetype	xsd:string	Yes	The access control entry types. See [RFC3530]
identifier	xsd:string	Yes	The user identifier, special "who"see [RFC3530]
aceflags	xsd:string	Yes	The semantics of the ACE. See [RFC3530]
acemask	xsd:string	Yes	The operations on a file or folder(directory in NFS v4) See [RFC3530]

5.2.2.7 Type: HashInformation

This type represents the file hash information.

Element	Type	Optional	Description
algorithm	xsd:string	No	The hash algorithm used (only "sha-1" currently supported). See [RFC5547].
value	xsd:hexBinary	No	The hash value of the file. See [RFC5547].

5.2.2.8 Type: Folder

Individual folder

Element	Type	Optional	[ResourceRelPath]	Description
folderAttributes	FolderAttributes	Yes	folderAttributes	List of attributes associated with the folder
cursor	xsd:string	Yes	Not applicable	If the lists of subfolders and files are both complete, this element is omitted. If there are more available subfolders and/or files not included in these lists, this element is included. The cursor value encapsulates information on these items. See section 6.1.3.3 for how to use the cursor in a subsequent request.
subFolders	ReferenceList	Yes	Not applicable	List of sub-folders under this folder. The client SHALL NOT include this element in PUT requests.
files	ReferenceList	Yes	Not applicable	List of files under this folder. The client SHALL NOT include this element in PUT requests.

resourceURL	xsd:anyURI	Yes	Not applicable	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named folder of type Folder is allowed in response bodies.

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.9 Type: FolderAttributes

Folder attributes.

Element	Type	Optional	[ResourceRelPath]	Description
root	xsd:boolean	Yes	Not applicable	The value “true” denotes the folder is designated as a root folder.
size	xsd:unsignedInt	Yes	Not applicable	The folder size. Indicates the size of the stored subfolders and files in bytes;
createTime	xsd:dateTimeStamp	Yes	Not applicable	Date and Time at which the folder was created.
filesNumber	xsd:unsignedInt	Yes	Not applicable	The number of files in this folder.
subFoldersNumber	xsd:unsignedInt	Yes	Not applicable	The number of sub folders in this folder.
owner	xsd:string	Yes	Not applicable	The owner of the folder, e.g. userId
accessControlList	AccessControlList	Yes	folderAttributes/acl	The access control list information. Which can be set by Update Folder ACL information operation.

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.10 Type: ReferenceList

List of object references

Element	Type	Optional	Description
reference	common:ResourceReference [0..unbounded]	Yes	A list of resource references

5.2.2.11 Type: DeleteMode

The delete mode

Element	Type	Optional	Description
deleteMode	DeleteModeEnum	No	The delete mode

A root element named deleteMode of type DeleteMode is allowed in request bodies.

5.2.2.12 Type: Share

The file sharing.

Element	Type	Optional	Description
isShare	xsd:boolean	No	Whether share file or not. Default is false.
shareLink	common: Link	Yes	The file share link
accessCode	xsd:string	Yes	The code to access the file via the link.

5.2.2.13 Type: ShareList

The file sharing list.

Element	Type	Optional	Description
share	Share[0..unbounded]	Yes	List of file sharing

A root element named shareList of type ShareList is allowed in response bodies.

5.2.2.14 Type: SearchCriteria

Search criteria.

Element	Type	Optional	Description
fromCursor	xsd:string	Yes	The beginning position of the retrieve response. Omitting this value denotes the first position. The fromCursor is a cursor value provided by the server in a previous response to a request with the same search selection criteria.

maxEntries	xsd:int	Yes	Specifies maximum number of entries to be returned in the response. Note: A server pre-defined (i.e., implementation specific) maximum number of entries MAY be returned in case the requested maximum exceeds server's pre-defined maximum entries.
searchKey	xsd:string	Yes	Search key If there is no search key, the server will retrieval all available elements.
searchScope	common:ResourceReference	Yes	Reference to folder at which point the search would start. If searchScope is provided, the scope of the search is limited to the subtree starting at this folder. If searchScope is not provided, the search is applied to the root folder.
sortCriterion	xsd:string	Yes	The sort criterion for the retrieval of elements. Default is random or server preferred sort.

A root element named searchCriteria of type SearchCriteria is allowed in request bodies.

5.2.2.15 Type: SearchResult

The search result

Element	Type	Optional	Description
result	Result	Yes	The search results. Number of results MAY be limited by the server.
cursor	xsd:string	Yes	If the list of results is complete, this element is omitted. If there are more available results not included in the list, then a cursor value is returned, which encapsulates information on these results. The client can use the cursor in a subsequent request, to hint to the server that it is asking for the rest of results which had not been returned in a previous request. The cursor encapsulates server state information which might be volatile, especially in a multi-device environment. Therefore the cursor mechanism makes no guarantee on the integral continuity of results returned in subsequent requests. The value and format of the string are implementation specific. Clients SHOULD NOT attempt to interpret or alter the cursor value.
resourceURL	xsd:anyURI	No	Self referring URL.

A root element named searchResult of type SearchResult is allowed in response bodies.

5.2.2.16 Type: Result

The search result information.

Element	Type	Optional	Description
folderSearchResult	FolderList	Yes	List of folders.
fileSearchResult	FileList	Yes	List of files.

5.2.2.17 Type: FolderList

The list of folder information.

Element	Type	Optional	Description
folder	Folder[0..unbounded]	Yes	A list of folder.

5.2.2.18 Type: FileList

The list of file information.

Element	Type	Optional	Description
file	File[0..unbounded]	Yes	A list of file.

5.2.2.19 Type: RecycleBin

List of Recycle Bin.

Element	Type	Optional	Description
recycleBinItemList	RecycleBinItemList	Yes	List of Recycle Bin
resourceURL	xsd:anyURI	No	Self referring URL.

A root element named recycleBin of type RecycleBin is allowed in request and/or response bodies.

5.2.2.20 Type: RecycleBinItemList

The list of metadata information.

Element	Type	Optional	Description
recycleBinItem	RecycleBinItem [0..unbounded]	Yes	A list of metadata.
recycleBinTreatment	RecycleBinTreatmentEnum	Yes	Recycle Bin treatment. The recycleBinTreatment SHALL NOT be included in GET responses by the Server. If there is no elements of recycleBinItem in PUT request, it meanings to clean the total Recycle Bin.(recycleBinTreatment value must be “Delete”)

5.2.2.21 Type: RecycleBinItem

The item in Recycle Bin.

Element	Type	Optional	Description
type	xsd:string	No	The Recycle Bin item type, value=0 meanings folder, value=1 meanings file
name	xsd:string	No	The folder or file name in Recycle Bin.
originalPath	xsd:string	Yes	The original path of folder or file before in Recycle Bin.
recycleBinItemAttributes	RecycleBinItemAttributes	Yes	Attributes associated with the file or folder in Recycle Bin.

5.2.2.22 Type: RecycleBinItemAttributes

The Recycle Bin item attributes.

Element	Type	Optional	Description
fileType	xsd:string	Yes	The file type, (i.e., jpg, doc, xls, zip). It is only used for RecycleBinitem type value=1 meanings file
size	xsd:unsignedInt	Yes	The item size.
deleteTime	xsd:dateTimeStamp	Yes	Date and Time at which the item was deleted.
createTime	xsd:dateTimeStamp	Yes	Date and Time at which the item was created.

5.2.3 Enumerations

The subsections of this section define the enumerations used in the UCD API.

5.2.3.1 Enumeration: DeleteModeEnum

Enumeration	Description
DeletePermanently	Remove from server storage and no revoke
DeleteToRecycleBin	Temporarily move to Recycle Bin and can revoke

5.2.3.2 Enumeration: RecycleBinTreatmentEnum

Enumeration	Description
Revoke	Revoke the Recycle Bin items.
Delete	Delete the Recycle Bin items.

5.2.4 Values of the Link “rel” attribute

The “rel” attribute of the Link element is a free string set by the server implementation, to indicate a relationship between the current resource and an external resource. The following are possible strings (list is non-exhaustive, and can be extended):

- FileShareLink

These values indicate the kind of resource that the link points to.

5.3 Sequence Diagrams

The following subsections describe the resources, methods and steps involved in typical scenarios.

5.3.1 Folder Operations

This figure below shows a scenario for creating, reading and deleting a folder.

The resources:

- To create, read and delete a folder, using resource under

  http://{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}

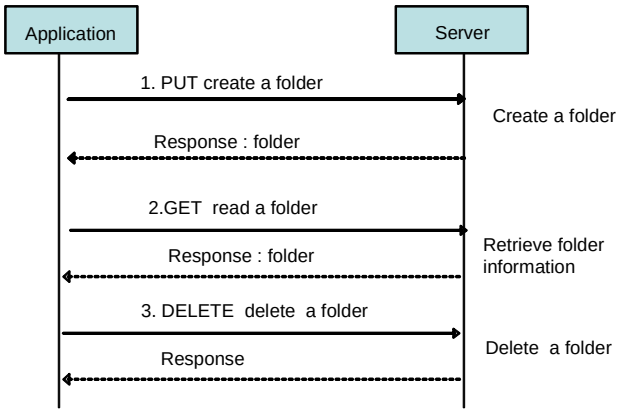

Figure 2 Folder Operations

Outline of the flows:

1. The application sends folder creation request using PUT method, the server responses with the result including folder information .
2. The application retrieves the information of a folder by using a GET method. The server responses with the result including folder information.
3. The application deletes a folder by using a DELETE method. The server deletes the folder permanently or to the recycle bin according to the delete mode parameter and responses with the result.

5.3.2 Folder attributes operations

This figure below shows a scenario for reading and updating the attributes.of a folder.

The resources:

- To get the attributes of a folder , the following Light-weight Resource is used
  http://{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/[ResourceRelPath]

  Where [ResourceRelPath] is a light-weight relative resource URL, and in this case it shall be replaced with "folderAttributes"

- To update access control list of a folder , the following Light-weight Resource is used
  http://{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/[ResourceRelPath]

  Where [ResourceRelPath] is a light-weight relative resource URL, and in this case it shall be replaced with "folderAttributes/acl"

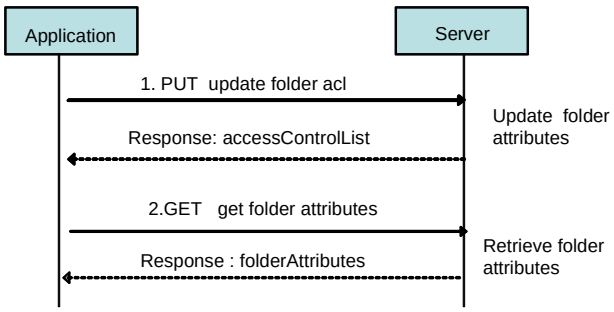

Figure 3 Folder attributes operations

Outline of the flows:

1. The application updates the access control list of a folder by using a PUT method on the Light-weight Resource for access control list of a folder. The server responses with the update result.
2. The application gets the attributes of a folder by using a GET method on the Light-weight Resource for folder attributes. The server responses with the folder attributes.

5.3.3 File operations

This figure below shows a scenario for creating, reading and deleting a file.

The resources:

- To create, read and delete a file, using the following resource:

  http://{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/{filename}

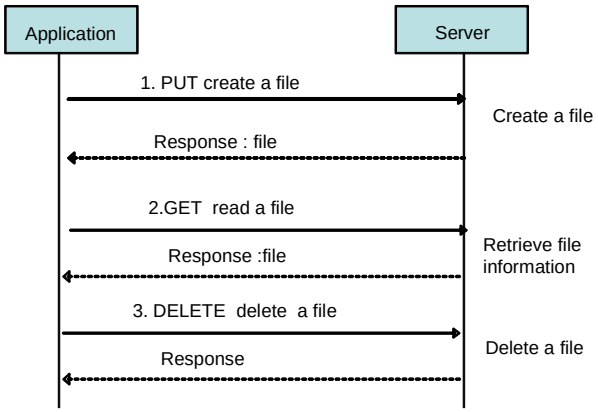

Figure 4 File Operations

Outline of the flows:

1. The application sends file creation request using PUT method. The server responses with the result including file information.
2. The application retrieves the information of a file by using a GET method. The server responses with the result including file information.
3. The application deletes a filer by using a DELETE method. The server deletes the file permanently or to the recycle bin according to the delete mode parameter and responses with the result.

5.3.4 File attributes operations

This figure below shows a scenario for reading and updating the attributes of a file.

The resources:

- To get the attributes of a file , the following Light-weight Resource is used
  http://{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/{filename}/[ResourceRelPath]

  Where [ResourceRelPath] is a light-weight relative resource URL, and in this case it shall be replaced with "fileAttributes"

- To update some attributes of a file, including filetype or user defined metadata or sharing option or access control list, the following Light-weight Resource is used
  http://{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/{filename}/[ResourceRelPath]

  Where [ResourceRelPath] is a light-weight relative resource URL, and in this case it shall be replaced with "fileAttributes/fileType" or "fileAttributes/metadata" or "fileAttributes/share" or "fileAttributes/acl"

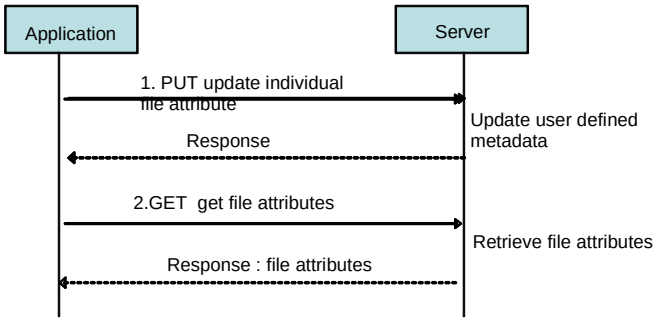

Figure 5 File attributes operations

Outline of the flows:

1. The application updates an individual attribute, e.g the metadata of a file by using a PUT method on the Light-weight Resource for metadata of a file. The server responses with the update result.
2. The application gets the attributes of a file by using a GET method on the Light-weight Resource for file attributes . The server responses with the file attributes.

5.3.5 Recycle bin operations

This figure below shows a scenario for listing, revoking or cleaning the recycle bin.

The resources:

- To list, revoke and delete the recycle bin , the following resource is used
  http://{serverRoot}/ucd/{apiVersion}/{userId}/{recycleBin}

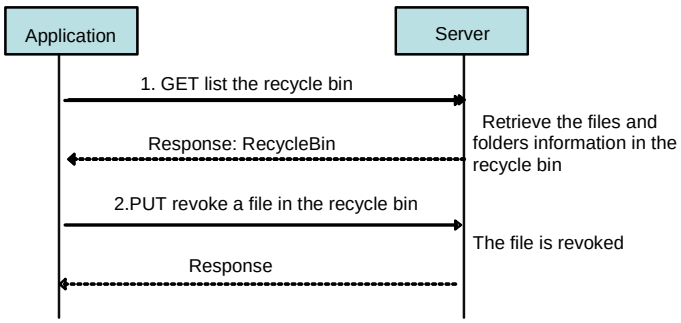

Figure 6 Recycle bin operations

Outline of the flows:

1. The application gets the recycle bin information by using a GET method, the server responses with the information of the list of files and folders in the recycle bin.
2. The application revokes/deletes files/folders in the recycle bin by using a PUT method, the server responses with the result.

5.3.6 Search operation

This figure below shows a scenario for searching files and file folders.

The resources:

- To search files and folders, the following resource is used

http://{serverRoot}/ucd/{apiVersion}/{userId}/operations/search

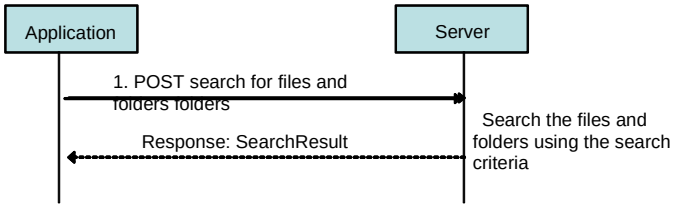

Figure 7 Search operation

1. The application searches the files and folders by using a POST method, with optional search criteria, the server responses with the search result.

5.3.7 List shared file operation

This figure below shows a scenario for listing the shared files of a user.

The resources:

- To list the shared files of a user, the following resource is used

http://{serverRoot}/ucd/{apiVersion}/{userId}/operations/listShare

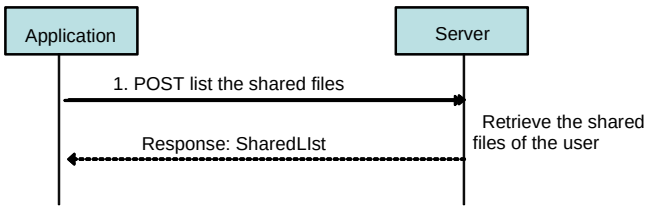

Figure 8 List shared file operation

1. The application gets the list of shared file by using a POST method, the server responses with the list of shared files of the user.

6. Detailed specification of the resources

The following applies to all resources defined in this specification regardless of the representation format (i.e. XML, JSON, application/x-www-form-urlencoded):

- Reserved characters in URL variables (parts of a URL denoted below by a name in curly brackets) MUST be percent-encoded according to [RFC3986]. Note that this always applies, no matter whether the URL is used as a Request URL or inside the representation of a resource (such as in "resourceURL" and "link" elements).
- If a user identifier (e.g. address, participantAddress, etc.) of type anyURI is in the form of an MSISDN, it MUST be defined as a global number according to [RFC3966] (e.g. tel:+19585550100). The use of characters other than digits and the leading "+" sign SHOULD be avoided in order to ensure uniqueness of the resource URL. This applies regardless of whether the user identifier appears in a URL variable or in a parameter in the body of an HTTP message.
- If an equipment identifier of type anyURI is in the form of a SIP URI, it MUST be defined according to [RFC3261].
- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an Anonymous Customer Reference (ACR), it MUST be defined according to [IETF_ACR_draft], i.e. it MUST include the protocol prefix 'acr:' followed by the ACR.
    - The ACR 'authorization' is a supported reserved keyword, and MUST NOT be assigned as an ACR to any particular end user. See G.1.2 for details regarding the use of this reserved keyword.
- For requests and responses that have a body, the following applies: in the requests received, the server SHALL support JSON and XML encoding of the parameters in the body, and MAY support application/x-www-form-urlencoded parameters in the body. The Server SHALL return either JSON or XML encoded parameters in the response body, according to the result of the content type negotiation as specified in [REST_NetAPI_Common]. In notifications to the Client, the server SHALL use either XML or JSON encoding, depending on which format the client has specified in the related subscription. The generation and handling of the JSON representations SHALL follow the rules for JSON encoding in HTTP Requests/Responses as specified in [REST_NetAPI_Common].

6.1 Resource: A folder

The resource used is:

//{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}

This resource is used for managing a folder such as listing folder information, creating folder, deleting a folder to recycle bin or permanently.

6.1.1 Request URL variables

The following request URL variables are common for all HTTP methods:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1
userId	Identifier of user.
folderName	The folder name which is absolute value including path. It starts from the root folder and ending with the given folder's name where the folder names are separated by a "/" (U+002F) character.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.1.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Unified Cloud Disk, see section 7.

6.1.3 GET

This operation is used for retrieval of a folder's properties and the list of contained subfolders and files.

Supported parameters in the query string of the Request URL are:

Name	Type/Values	Optional	Description
fromCursor	xsd:string	Yes	The beginning position of the retrieve response. Omitting this value denotes the first position. The fromCursor is a cursor value provided by the server in a previous response to a request for the same folder;
maxEntries	xsd:int	Yes	Specifies maximum number of entries to be returned in the response. The server MAY return fewer entries than this. Default is provided by server policy.

6.1.3.1 Example 1: Retrieve information about a folder (Informative)

6.1.3.1.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```

6.1.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
  <ucd:folder xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
    <folderAttributes>
      <root>false</root>
      <size>5030248</size>
      <createTime>2014-01-19T08:30:50Z</createTime>
      <filesNumber>2</filesNumber>
      <subFoldersNumber>1</subFoldersNumber>
      <owner>George Smith</owner>
     <accessControlList>
       <accessControlEntry>
         <acetype>ALLOW</acetype>
         <identifier>OWNER@</identifier>
         <aceflags>DIRECTORY_INHERIT</aceflags>
         <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE </acemask>
```

```
      </accessControlEntry>
    </accessControlList>
  </folderAttributes>
  <subFolders>
    <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel</resourceURL>
</reference>    </subFolders>
   <files>
    <reference>
      <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc</r
        esourceURL>
    </reference>
    <reference>
      <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</re
        sourceURL>
</reference>   </files>
  <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument</resourceURL>
</ucd:folder>
```

6.1.3.2 Example 2: Retrieve information about a non-existent folder (Informative)

6.1.3.2.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fotherdocument  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```

6.1.3.2.2 Response

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Fri, 17 Jan 2014 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <link rel="folder"
    href="http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fotherdocument"/>
  <serviceException>
    <messageId>SVC0004</messageId>
    <text>No valid addresses provided in message part %1</text>
    <variables>Request-URI</variables>
  </serviceException>
</common:requestError>
```

6.1.3.3 Example 3: Retrieve information about a large folder (Informative)

6.1.3.3.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument?maxEntries=1  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```


6.1.3.3.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:folder xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <folderAttributes>
      <root>false</root>
      <size>5030248</size>
      <createTime>2014-01-19T08:30:50Z </createTime>
      <filesNumber>2</filesNumber>
      <subFoldersNumber>1</subFoldersNumber>
      <owner>George Smith</owner>
     <accessControlList>
       <accessControlEntry>
          <acetype>ALLOW</acetype>
          <identifier>OWNER@</identifier>
          <aceflags>DIRECTORY_INHERIT</aceflags>
          <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE </acemask>
       </accessControlEntry>
     </accessControlList>
  </folderAttributes>
 <cursor>cursor111</cursor>
 <subFolders>    <reference>
<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel</resourceURL>
</reference>
  </subFolders>
 <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument</resourceURL>
</ucd:folder>
```

6.1.3.4 Example 4: Retrieve information about a large folder (Informative)

This example continues the previous one, by passing back the cursor provided by the server.

6.1.3.4.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument?fromCursor=abcdef%3Fcur%38194&maxEntries=2
HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```

6.1.3.4.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:folder xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <folderAttributes>
```

```
      <root>false</root>
      <size>5030248</size>
       <createTime>2014-01-19T08:30:50Z </createTime>
      <filesNumber>2</filesNumber>
      <subFoldersNumber>1</subFoldersNumber>
      <owner>George Smith</owner>
     <accessControlList>
       <accessControlEntry>
          <acetype>ALLOW</acetype>
          <identifier>OWNER@</identifier>
          <aceflags>DIRECTORY_INHERIT</aceflags>
          <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE </acemask>
       </accessControlEntry>
     </accessControlList>
  </folderAttributes>
  <files>   <reference>
<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc</resour
ceURL>    </reference>    <reference>
<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</resourc
eURL>    </reference>
  </files>
  <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument</resourceURL>
</ucd:folder>
```

6.1.4 PUT

This operation is used for creating a new folder.

6.1.4.1 Example 1: Create folder, response with a location of created resource (Informative)

The following example shows a request for creating a new folder called “mydocument” to be created under the folder with path “/myfolder”.This example assumes that a folder with path “/myfolder” already exists.

6.1.4.1.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument HTTP/1.1
Accept: application/xml
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
```

6.1.4.1.2 Response

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>

<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL> http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument</resourceURL>
```



```
</common:resourceReference>
```

6.1.4.2 Example 2: Create folder, response with a copy of created resource (Informative)

The following example shows a request for creating a new folder called “mydocument” to be created under the folder with path “/myfolder”.This example assumes that a folder with path “/myfolder” already exists.

6.1.4.2.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument HTTP/1.1
Accept: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

6.1.4.2.2 Response

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:folder xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <folderAttributes>
      <root>false</root>
      <size>0</size>
      <createTime>2014-01-19T08:30:50Z</createTime>
      <filesNumber>0</filesNumber>
      <subFoldersNumber>0</subFoldersNumber>
      <owner>George Smith</owner>
     <accessControlList>
       <accessControlEntry>
          <acetype>ALLOW</acetype>
          <identifier>OWNER@</identifier>
          <aceflags>DIRECTORY_INHERIT</aceflags>
          <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE </acemask>
       </accessControlEntry>
     </accessControlList>
  </folderAttributes>
  <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument</resourceURL>
</ucd:folder>
```

6.1.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: GET, DELETE’ field in the response as per section 14.7 of [RFC2616].

6.1.6 DELETE

This operation is used to delete a folder to recycle bin or permanently. All the contained subfolders and files in the targeted folder SHALL be deleted as well.

The server responds to a DELETE request with an HTTP 204 No Content response.

6.1.6.1 Example 1: Delete a folder to recycle bin, response with "204 No Content" (Informative)

6.1.6.1.1 Request

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument  HTTP/1.1
Host: example.com
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<ucd:deleteMode xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <deleteMode>DeleteToRecycleBin</deleteMode>
</ucd:deleteMode>
```

6.1.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

6.1.6.2 Example 2: Delete a folder permanently, response with "204 No Content" (Informative)

6.1.6.2.1 Request

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument  HTTP/1.1
Host: example.com
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<ucd:deleteMode xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <deleteMode>DeletePermanently</deleteMode>
</ucd:deleteMode>
```

6.1.6.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

6.2 Resource: Individual folder attributes

The resource used is:

//{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/[ResourceRelPath]

This resource is used to retrieve individual folder attributes or update individual folder ACL information.

6.2.1 Request URL variables

The following request URL variables are common for all HTTP methods:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1
userId	Identifier of user.
folderName	The folder name which is absolute value including path. It starts from the root folder and ending with the given folder's name where the folder names are separated by a "/" (U+002F) character.
[ResourceRelPath]	Relative resource path for a Light-weight Resource, consisting of a relative path down to an element in the data structure. For more information about the applicable values (strings) for this variable, see 6.2.1.1.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.2.1.1 Light-weight relative resource paths

The following table describes the type of Light-weight Resources that can be accessed by using this resource, applicable methods, and the link to a data structure that contains values (strings) for those relative resource paths.

Light-weight Resource type	Method supported	Description
Individual folder attributes	GET	Enables retrieve individual folder attributes. See column [ResourceRelPath] for element "folderAttributes" in section 5.2.2.8 for possible values for the Light-weight relative resource path.
individual folder ACL information	PUT	Enables update individual folder ACL information. See column [ResourceRelPath] for element "folderAttributes/acl" in section 5.2.2.9 for possible values for the Light-weight relative resource path.

6.2.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Unified Cloud Disk, see section 7.

6.2.3 GET

This operation is used for retrieval individual folder attributes.

6.2.3.1 Example: Retrieve a folder's attributes (Informative)

6.2.3.1.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/folderAttributes
HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```

6.2.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2012 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:folderAttributes xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
      <root>false</root>
      <size>5030248</size>
      <createTime>2014-01-19T08:30:50Z</createTime>
      <filesNumber>2</filesNumber>
      <subFoldersNumber>1</subFoldersNumber>
      <owner>George Smith</owner>
      <accessControlList>
        <accessControlEntry>
          <acetype>ALLOW</acetype>
          <identifier>OWNER@</identifier>
          <aceflags>DIRECTORY_INHERIT</aceflags>
          <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE </acemask>
        </accessControlEntry>
      </accessControlList>
</ucd:folderAttributes>
```

6.2.4 PUT

This operation is used to update individual folder ACL information.

6.2.4.1 Example 1: Update individual folder ACL information (Informative)

6.2.4.1.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/folderAttributes/acl HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:accessControlList xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1"><accessControlEntry>
  <acetype>ALLOW</acetype>
  <identifier>OWNER@</identifier>
  <aceflags>DIRECTORY_INHERIT</aceflags>
  <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD</acemask>
 </accessControlEntry>
</ucd:accessControlList>
```

6.2.4.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ucd:accessControlList xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <accessControlEntry><acetype>ALLOW</acetype>
    <identifier>OWNER@</identifier>
    <aceflags>DIRECTORY_INHERIT</aceflags>
    <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD</acemask>
  </accessControlEntry>
</ucd:accessControlList>
```

6.2.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT' field in the response as per section 14.7 of [RFC2616].

6.2.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT' field in the response as per section 14.7 of [RFC2616].

6.3 Resource: A file

The resource used is:

//{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/{fileName}

This resource is used for managing a file such as downloading a file, uploading a file, updating file in range with the start byte and the end byte, deleting a file to recycle bin or permanently.

6.3.1 Request URL variables

The following request URL variables are common for all HTTP methods:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1
userId	Identifier of user.
folderName	The folder name which is absolute value including path. It starts from the root folder and ending with the given folder's name where the folder names are separated by a "/" (U+002F) character.
fileName	The file name.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.3.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Unified Cloud Disk, see section 7.

6.3.3 GET

This operation is used for downloading a file.

6.3.3.1 Example 1: Downloading a file (Informative)

6.3.3.1.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```

6.3.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Length: nnnn

[mydocument1.doc's content]
```

6.3.4 PUT

This operation is used for uploading a file or updating file in range with the start byte and the end byte.

6.3.4.1 Example 1: Uploading a file, response with a location of created resource (Informative)

The following example shows a request for uploading a new file called "mydocument2.jpg" to the folder "/myfolder/mydocument". This example assumes that a folder "/myfolder/mydocument" already exists.

6.3.4.1.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg HTTP/1.1
Accept: application/xml
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
Content-Type: image/jpg
Content-Length: nnnn
MIME-Version: 1.0

[mydocument2.jpg's content]
```

6.3.4.1.2 Response

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</resourceURL>
</common:resourceReference>
```

6.3.4.2 Example 2: Uploading a file, response with a copy of created resource (Informative)

The following example shows a request for uploading a new file called “mydocument2.jpg” to the folder “/myfolder/mydocument”. This example assumes that a folder “/myfolder/mydocument” already exists.

6.3.4.2.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg HTTP/1.1
Accept: application/xml
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
Content-Type: image/jpg
Content-Length: nnnn
MIME-Version: 1.0

[mydocument2.jpg’s content]
```

6.3.4.2.2 Response

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:file xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
<fileAttributes>
  <fileType>jpg</fileType>
  <size>598208</size>
  <createTime>2014-01-19T08:30:50Z</createTime>
  <owner>George Smith</owner>
  <metadataList>
    <metadata>
      <name>publisher</name>
      <value>XYZ</value>
    </metadata>
    <metadata>
      <name>project</name>
      <value>abc</value>
    </metadata>
    <metadata>
      <name>department</name>
      <value>Sales</value>
    </metadata>
  </metadataList>
  <accessControlList>
    <accessControlEntry>
    <acetype>ALLOW</acetype>
    <identifier>OWNER@</identifier>
    <aceflags>FILE_INHERIT</aceflags>
    <acemask>READ_DATA, WRITE_DATA, APPEND_DATA, READ_NAMED_ATTRS, WRITE_NAMED_ATTRS,
READ_ATTRIBUTES, WRITE_ATTRIBUTES, DELETE, READ_ACL, WRITE_ACL, WRITE_OWNER,
SYNCHRONIZE</acemask></accessControlEntry>
  </accessControlList>
```

```
  <hash>
    <algorithm>sha-1</algorithm>
    <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
  </hash>
  <share>
    <isShare>false</isShare>
  </share>
  <revisionList>
   <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg.rev001</
resourceURL>
    </reference>
  </revisionList>
 </fileAttributes>

<resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</resourceURL></ucd:f
ile>
```

6.3.4.3 Example 3: Updating file in range, response with a copy of created resource (Informative)

The following example shows a request for updating file called “mydocument1.doc” in range with the start byte and the end byte to the folder “/myfolder/mydocument”. This example assumes that a folder “/myfolder/mydocument” already exists.

6.3.4.3.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc HTTP/1.1
Accept: application/xml
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
Content-Type: binary/oct-stream
Range:bytes=start-end

[mydocument1.doc’s updating content]
```

6.3.4.3.2 Response

```
HTTP/1.1 200 OK
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc
Content-Type: application/xml
Content-Range: start-end/size

<?xml version="1.0" encoding="UTF-8"?>
<ucd:file xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
<fileAttributes>
  <fileType>doc</fileType>
  <size>230208</size>
  <createTime>2014-01-09T18:20:40Z</createTime>
  <modifyTime>2014-01-19T08:30:50Z</modifyTime>
  <accessTime>2014-01-10T12:10:30Z</accessTime>
  <owner>George Smith</owner>
  <metadataList>
    <metadata>
```

```
      <name>publisher</name>
      <value>HZ</value>
    </metadata>
    <metadata>
      <name>project</name>
      <value>abc</value>
    </metadata>
    <metadata>
      <name>department</name>
      <value>Sales</value>
    </metadata>
  </metadataList>
  <accessControlList><accessControlEntry>
    <acetype>ALLOW</acetype>
    <identifier>OWNER@</identifier>
    <aceflags>DIRECTORY_INHERIT</aceflags>
    <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL, WRITE_OWNER,
SYNCHRONIZE </acemask></accessControlEntry>
  </accessControlList>
  <hash>
    <algorithm>sha-1</algorithm>
    <value>86D471913EE4B1DF2F58231FE8653BBCF371362F</value>
  </hash>
  <share>
    <isShare>false</isShare>
  </share>
  </fileAttributes>
  <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc</resourceURL>
</ucd:file>
```

6.3.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC2616].

6.3.6 DELETE

This operation is used to delete a file to recycle bin or permanently.

The server responds to a DELETE request with an HTTP 204 No Content response.

6.3.6.1 Example 1: Delete a file to recycle bin, response with "204 No Content" (Informative)

6.3.6.1.1 Request

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc  HTTP/1.1
Host: example.com
Accept: application/xml


<?xml version="1.0" encoding="UTF-8"?>
<ucd:deleteMode xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <deleteMode>DeleteToRecycleBin</deleteMode>
</ucd:deleteMode>
```


6.3.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

6.3.6.2 Example 2: Delete a file permanently, response with "204 No Content" (Informative)

6.3.6.2.1 Request

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc  HTTP/1.1
Host: example.com
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<ucd:deleteMode xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <deleteMode>DeletePermanently</deleteMode>
</ucd:deleteMode>
```

6.3.6.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

6.4 Resource: Individual file attributes

The resource used is:

//{serverRoot}/ucd/{apiVersion}/{userId}/{folderName}/{fileName}/[ResourceRelPath]

This resource is used to retrieve individual file attributes or update individual file attributes including fileType, metadata, ACL information, sharing information.

6.4.1 Request URL variables

The following request URL variables are common for all HTTP methods:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1
userId	Identifier of user.
folderName	The folder name which is absolute value including path. It starts from the root folder and ending with the given folder's name where the folder names are separated by a "/" (U+002F) character.
fileName	The file name.

[ResourceRelPath]	Relative resource path for a Light-weight Resource, consisting of a relative path down to an element in the data structure. For more information about the applicable values (strings) for this variable, see 6.4.1.1.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.4.1.1 Light-weight relative resource paths

The following table describes the type of Light-weight Resources that can be accessed by using this resource, applicable methods, and the link to a data structure that contains values (strings) for those relative resource paths.

Light-weight Resource type	Method supported	Description
Individual file attributes	GET	Enables retrieve individual folder attributes. See column [ResourceRelPath] for element “folderAttributes” in section 5.2.2.1 for possible values for the Light-weight relative resource path.
individual file attributes including fileType, metadata, ACL information, sharing information	PUT	Enables update file attributes including fileType, metadata, ACL information, sharing information. See column [ResourceRelPath] for element “folderAttributes/acl” in section 5.2.2.2 for possible values for the Light-weight relative resource path.

6.4.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Unified Cloud Disk, see section 7.

6.4.3 GET

This operation is used for retrieval individual file attributes.

6.4.3.1 Example: Retrieve a file’s attributes (Informative)

6.4.3.1.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes
HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```

6.4.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2012 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:fileAttributes xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <fileType>jpg</fileType>
  <size>598208</size>
  <createTime>2014-01-19T08:30:50Z</createTime>
  <owner>George Smith</owner>
```

```
  <metadataList>
    <metadata>
      <name>publisher</name>
      <value>XYZ</value>
    </metadata>
    <metadata>
      <name>project</name>
      <value>abc</value>
    </metadata>
    <metadata>
      <name>department</name>
      <value>Sales</value>
    </metadata>
  </metadataList>
  <accessControlList>
      <accessControlEntry>
      <acetype>ALLOW</acetype>
      <identifier>OWNER@</identifier>
      <aceflags>DIRECTORY_INHERIT</aceflags>
      <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE </acemask>
    </accessControlEntry>
  </accessControlList>
  <hash>
    <algorithm>sha-1</algorithm>
    <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
  </hash>
  <share>
    <isShare>false</isShare>
  </share>
  <revisionList>
   <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg.rev001</
resourceURL>
    </reference>
  </revisionList>
</ucd:fileAttributes>
```

6.4.4 PUT

This operation is used to update individual file attributes including fileType, metadata, ACL information, sharing information.

6.4.4.1 Example 1: Update individual file's attribute of fileType (Informative)

6.4.4.1.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/fileType HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn
```



```
<?xml version="1.0" encoding="UTF-8"?>
<ucd:fileType xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">png</ucd:fileType>
```

6.4.4.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:fileType xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">png</ucd:fileType>
```

6.4.4.2 Example 2: Update individual file's attribute of metadata (Informative)

6.4.4.2.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/metadata HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:metadataList xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
 <metadata>
  <name>publisher</name>
  <value>XYZ</value>
 </metadata>
 <metadata>
  <name>project</name>
  <value>def</value>
 </metadata>
 <metadata>
  <name>department</name>
  <value>Sales</value>
 </metadata>
</ucd:metadataList>
```

6.4.4.2.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:metadataList xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
 <metadata>
  <name>publisher</name>
  <value>XYZ</value>
 </metadata>
 <metadata>
  <name>project</name>
```

```
   <value>def</value>
  </metadata>
  <metadata>
   <name>department</name>
   <value>Sales</value>
  </metadata>
</ucd:metadataList>
```

6.4.4.3 Example 3: Update individual file ACL information (Informative)

6.4.4.3.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/acl HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:accessControlList xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <accessControlEntry>
    <acetype>ALLOW</acetype>
    <identifier>OWNER@</identifier>
    <aceflags>FILE_INHERIT</aceflags>
    <acemask>READ_DATA, WRITE_DATA, APPEND_DATA, DELETE, READ_ACL, WRITE_ACL, WRITE_OWNER,
SYNCHRONIZE</acemask>
  </accessControlEntry>
</ucd:accessControlList>
```

6.4.4.3.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:accessControlList xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <accessControlEntry>
    <acetype>ALLOW</acetype>
    <identifier>OWNER@</identifier>
    <aceflags>FILE_INHERIT</aceflags>
    <acemask>READ_DATA, WRITE_DATA, APPEND_DATA, DELETE, READ_ACL, WRITE_ACL, WRITE_OWNER,
SYNCHRONIZE</acemask>
  </accessControlEntry>
</ucd:accessControlList>
```

6.4.4.4 Example 4: Update individual file sharing information (Informative)

6.4.4.4.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/share HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```



```
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>

<ucd:share xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <isShare>true</isShare>
  <shareLink rel="FileShareLink"  href="http://example.com/GeorgeSmith/mydocument2.jpg"/>
  <accessCode>12345</accessCode>
</ucd:share>
```

6.4.4.4.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>

<ucd:share xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <isShare>true</isShare>
  <shareLink rel="FileShareLink"  href="http://example.com/GeorgeSmith/mydocument2.jpg"/>
  <accessCode>12345</accessCode>
</ucd:share>
```

6.4.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT' field in the response as per section 14.7 of [RFC2616].

6.4.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT' field in the response as per section 14.7 of [RFC2616].

6.5 Resource: RecycleBin

The resource used is:

//{serverRoot}/ucd/{apiVersion}/{userId}/recycleBin

This resource is used for managing recycle bin such as listing recycle bin, revoking or cleaning recycle bin.

6.5.1 Request URL variables

The following request URL variables are common for all HTTP methods:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI

apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1
userId	Identifier of user.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.5.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Unified Cloud Disk, see section 7.

6.5.3 GET

This operation is used for listing recycle bin.

Supported parameters in the query string of the Request URL are:

Name	Type/Values	Optional	Description
fromCursor	xsd:string	Yes	The beginning position of the retrieve response. Omitting this value denotes the first position. The fromCursor is a cursor value provided by the server in a previous response to a request for the same folder;
maxEntries	xsd:int	Yes	Specifies maximum number of entries to be returned in the response. The server MAY return fewer entries than this. Default is provided by server policy.

6.5.3.1 Example 1: List recycle bin (Informative)

6.5.3.1.1 Request

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/recycleBin  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
```

6.5.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:recycleBin xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
<recycleBinItemList>
  <recycleBinItem>
    <type>0</type>
    <name>mypicture</name>
    <originalPath>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmypicture</originalPath>
    <recycleBinItemAttributes>
      <size>51198208</size>
      <deleteTime>2014-02-01T07:29:45Z</deleteTime>
      <createTime>2014-01-19T08:30:50Z</createTime>
    </recycleBinItemAttributes>
```

```
  </recycleBinItem>
  <recycleBinItem>
   <type>1</type>
   <name>novel111.pdf</name>

<originalPath>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel111.pdf</orig
inalPath>
    <recycleBinItemAttributes>
     <fileType>pdf</fileType>
     <size>2030248</size>
     <deleteTime>2014-01-03T07:29:45Z</deleteTime>
     <createTime>2014-01-01T08:30:50Z</createTime>
    </recycleBinItemAttributes>
  </recycleBinItem>
 </recycleBinItemList>
 <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin</resourceURL>
</ucd:recycleBin>
```

6.5.4 PUT

This operation is used for revoking or deleting items in recycle bin.

6.5.4.1 Example 1: Revoking recycle bin items (Informative)

6.5.4.1.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/recycleBin HTTP/1.1
Host: example.com
Accept: application/xml


<?xml version="1.0" encoding="UTF-8"?>
<ucd:recycleBin xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
<recycleBinItemList>
  <recycleBinItem>
   <type>1</type>
   <name>novel111.pdf</name>

<originalPath>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel111.pdf</orig
inalPath>
    <recycleBinItemAttributes>
     <fileType>pdf</fileType>
     <size>2030248</size>
     <deleteTime>2014-01-03T07:29:45Z</deleteTime>
     <createTime>2014-01-01T08:30:50Z</createTime>
    </recycleBinItemAttributes>
   </recycleBinItem>
   <recycleBinTreatment>Revoke</recycleBinTreatment>
  </recycleBinItemList>
 <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin</resourceURL>
</ucd:recycleBin>
```

6.5.4.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 05 May 2014 06:05:09 GMT
```

6.5.4.2 Example 2: Clean the recycle bin (Informative)

6.5.4.2.1 Request

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/recycleBin HTTP/1.1
Host: example.com
Accept: application/xml


<?xml version="1.0" encoding="UTF-8"?>
<ucd:recycleBin xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <recycleBinItemList>
   <recycleBinTreatment>Delete</recycleBinTreatment>
  </recycleBinItemList>
 <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin</resourceURL>
</ucd:recycleBin>
```

6.5.4.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 05 May 2014 06:05:09 GMT
```

6.5.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC2616].

6.5.6 DELETE

This operation is used to delete a folder to recycle bin or permanently. All the contained subfolders and files in the targeted folder SHALL be deleted as well.

The server responds to a DELETE request with an HTTP 204 No Content response.

6.6 Resource: Search for files or folders

The resource used is:

//{serverRoot}/ucd/{apiVersion}/{userId}/operations/search

This resource is used to search folders or files.

6.6.1 Request URL variables

The following request URL variables are common for all HTTP methods:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1
userId	Identifier of user.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.6.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Unified Cloud Disk, see section 7.

6.6.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.6.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.6.5 POST

This operation is used for search folders or files, where the set is defined by selection criteria.

6.6.5.1 Example 1: Search (Informative)

6.6.5.1.1 Request

```
POST /exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/search HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:searchCriteria xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <maxEntries>10</maxEntries>
  <searchKey>*my*</searchKey>
  <searchScope>
    <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder</resourceURL>
  </searchScope>
  <sortCriterion>Ascending by Date</sortCriterion>
</ucd:searchCriteria>
```

6.6.5.1.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 07:51:50 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:searchResult xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
 <result>
  <folderSearchResult>
   <folder>
    <folderAttributes>
     <root>false</root>
     <size>15649778</size>
     <createTime>2014-01-05T06:03:05Z</createTime>
     <filesNumber>3</filesNumber>
     <subFoldersNumber>0</subFoldersNumber>
     <owner>George Smith</owner>
     <accessControlList>
      <accessControlEntry>
       <acetype>ALLOW</acetype>
       <identifier>OWNER@</identifier>
       <aceflags>DIRECTORY_INHERIT</aceflags>
       <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE </acemask>
      </accessControlEntry>
     </accessControlList>
    </folderAttributes>
    <files>
     <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel001.pdf</re
sourceURL>
     </reference>
     <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel002.pdf</re
sourceURL>
     </reference>
     <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel003.pdf</re
sourceURL>
     </reference>
    </files>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel</resourceURL>
   </folder>
  </folderSearchResult>
  <fileSearchResult>
   <file>
    <fileAttributes>
     <fileType>doc</fileType>
     <size>246918</size>
     <createTime>2014-01-09T18:20:40Z</createTime>
```

```
    <owner>George Smith</owner>
    <hash>
      <algorithm>sha-1</algorithm>
      <value>86D471913EE4B1DF2F58231FE8653BBCF371362F</value>
    </hash>
    <share>
     <isShare>false</isShare>
    </share>
   </fileAttributes>
   <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc</resourceURL>
  </file>
  <file>
   <fileAttributes>
    <fileType>jpg</fileType>
    <size>549778</size>
    <createTime>2014-01-19T08:30:50Z</createTime>
    <owner>George Smith</owner>
    <hash>
      <algorithm>sha-1</algorithm>
      <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
    </hash>
    <share>
     <isShare>true</isShare>

      <shareLink rel="FileShareLink"  href="http://example.com/GeorgeSmith/mydocument2.jpg"/>
     <accessCode>12345</accessCode>
    </share>
    <revisionList>

<reference><resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.j
pg.rev001</resourceURL>
       </reference>
     </revisionList>
    </fileAttributes>
    <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</resourceURL>
   </file>
  </fileSearchResult>
 </result>
 <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/search</resourceURL>
</ucd:searchResult>
```

6.6.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.7 Resource: List the shared files

The resource used is:

//{serverRoot}/ucd/{apiVersion}/{userId}/operations/listShare

This resource is used to list all file sharing.

6.7.1 Request URL variables

The following request URL variables are common for all HTTP methods:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1
userId	Identifier of user.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.7.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Unified Cloud Disk, see section 7.

6.7.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.7.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.7.5 POST

This operation is used to list all file sharing.

6.7.5.1 Example 1: List the shared files (Informative)

6.7.5.1.1 Request

```
POST /exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/listShare HTTP/1.1
Accept: application/xml
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
```

6.7.5.1.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 07:51:50 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:shareList xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
  <share>
    <isShare>true</isShare>
    <shareLink rel="FileShareLink"  href="http://example.com/GeorgeSmith/mydocument2.jpg"/>
```

```
   <accessCode>12345</accessCode>
  </share>
</ucd:shareList>
```

6.7.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

7. Fault definitions

7.1 Service Exceptions

For common Service Exceptions refer to [REST_NetAPI_Common]. There are no additional Service Exception codes defined for the RESTful Unified Cloud Disk API.

7.2 Policy Exceptions

For common Policy Exceptions refer to [REST_NetAPI_Common]. There are no additional Service Exception codes defined for the RESTful Unified Cloud Disk API.

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft Versions: OMA-TS-REST_NetAPI_UCD-V1_0	05 may 2014	All	Initial baseline OMA-CD-UCD-2014-0027-INP_REST_NetAPI_UCD_TS_base
	12 Jun 2014	All	OMA-CD-UCD-2014-0029R03-CR_UCD_2_resource_datatype_fix
	12 Jun 2014	section 6	OMA-CD-UCD-2014-0031R03-CR_UCD_2_example
	12 Jun 2014	section Appendix D	OMA-CD-UCD-2014-0033R02-CR_UCD_2_JSON_example
	12 Jun 2014	section Appendix C	OMA-CD-UCD-2014-0034-CR_UCD_2_x_www_form_urlencoded_example
	12 Jun 2014	section Appendix B	OMA-CD-UCD-2014-0035R01-CR_UCD_2_Static_Conformance_Requirements
	12 Jun 2014	section Appendix F	OMA-CD-UCD-2014-0036-CR_UCD_2_Light_weight_Resources
	12 Jun 2014	section 5.3	OMA-CD-UCD-2014-0039R01-CR_UCD_2_Sequence_Diagrams
	12 Jun 2014	section G	OMA-CD-UCD-2014-0042R02-CR_Interface_UCD_2_Authorization_Aspects
	09 Sep 2014	All	OMA-CD-UCD-2014-0082R01-CR_TS_REST_ZTE
	23 Sep 2014	section 6	OMA-CD-UCD-2014-0095-CR_TS_REST_ZTE
Candidate Version: OMA-TS-REST_NetAPI_UCD-V1_0	16 Dec 2014	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2014-0277-INP_UCD_V1_0_ERP_and_ETR_for_Candidate_Approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

B.1 SCR for REST.UCD Server

Item	Function	Reference	Requirement
REST-UCD-SUPPORT-S-001-M	Support for the RESTful UCD API	5, 6	
REST-UCD-SUPPORT-S-002-M	Support for the XML request & response format	6	
REST-UCD-SUPPORT-S-003-M	Support for the JSON request & response format	6	
REST-UCD-SUPPORT-S-004-O	Support for the application/x-www-form-urlencoded format	Appendix C	

B.1.1 SCR for REST.UCD.Folder Server

Item	Function	Reference	Requirement
REST-UCD-FOLDER-001-M	Support for folder operations	6.1	
REST-UCD-FOLDER-002-M	List folder information - GET	6.1.3	
REST-UCD-FOLDER-003-M	Create folder - POST	6.1.4	
REST-UCD-FOLDER-004-M	Delete folder to recycle bin or permanently - DELETE	6.1.6	

B.1.2 SCR for REST.UCD.Folder.Attr Server

Item	Function	Reference	Requirement
REST-UCD-FOLDER-ATTR-001-O	Support for the management of individual folder attributes	6.2	
REST-UCD-FOLDER-ATTR-002-O	Retrieve individual folder attributes- GET	6.2.3	
REST-UCD-FOLDER-ATTR-003-O	Update individual folder ACL information - PUT	6.2.4	

B.1.3 SCR for REST.UCD.File Server

Item	Function	Reference	Requirement
REST-UCD-FILE-001-M	Support for file operations	6.3	
REST-UCD-FILE-002-M	Download file - GET	6.3.3	
REST-UCD-FILE-003-M	Upload file or Update file in range - PUT	6.3.4	

Item	Function	Reference	Requirement
REST-UCD-FILE-004-M	Delete file to recycle bin or permanently- DELETE	6.3.6	

B.1.4 SCR for REST.UCD.File.Attr Server

Item	Function	Reference	Requirement
REST-UCD-FILE-ATTR-001-O	Support for the management of individual file attributes	6.4	
REST-UCD-FILE-ATTR-002-O	Retrieve individual file attributes- GET	6.4.3	
REST-UCD-FILE-ATTR-003-O	Update individual file information including fileType, metadata, ACL information, sharing information - PUT	6.4.4	

B.1.5 SCR for REST.UCD.Recyclebin Server

Item	Function	Reference	Requirement
REST-UCD-RECYCLEBIN-001-M	Support for Recycle bin operations	6.5	
REST-UCD-RECYCLEBIN-002-M	List recycle bin - GET	6.5.3	
REST-UCD-RECYCLEBIN-003-M	Revoke or Delete the items of recycle bin- PUT	6.5.4	

B.1.6 SCR for REST.UCD.Search Server

Item	Function	Reference	Requirement
REST-UCD-SEARCH-001-M	Support for search operations	6.6	
REST-UCD-SEARCH-002-M	Search file or folder - POST	6.6.5	

B.1.7 SCR for REST.UCD.ListShare Server

Item	Function	Reference	Requirement
REST-UCD-LISTSHARE-001-M	Support for listShare operations	6.7	
REST-UCD-LISTSHARE-002-M	List all file sharing - POST	6.7.5	

Appendix C. Application/x-www-form-urlencoded Request Format for POST Operations (Normative)

This section defines a format for the RESTful UCD API requests where the body of the request is encoded using the application/x-www-form-urlencoded MIME type.

Note: only the request body is encoded as application/x-www-form-urlencoded, the response is still encoded as XML or JSON depending on the preference of the client and the capabilities of the server. Names and values MUST follow the application/x-www-form-urlencoded character escaping rules from [W3C_URLENC].

The encoding is defined below for the following UCD REST operations which are based on POST requests:

- search folders or files

C.1 Search folders or files

This operation is used to search folders or files. See section 6.6.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
fromCursor	xsd:string	Yes	The beginning position of the retrieve response. Omitting this value denotes the first position. The fromCursor is a cursor value provided by the server in a previous response to a request with the same search selection criteria.
maxEntries	xsd:int	Yes	Specifies maximum number of entries to be returned in the response. Note: A server pre-defined (i.e., implementation specific) maximum number of entries MAY be returned in case the requested maximum exceeds server's pre-defined maximum entries.
searchKey	xsd:string	Yes	Search key If there is no search key, the server will retrieval all available elements.
resourceURL	xsd:anyURI	No	The URL that addresses the resource. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

sortCriterion	xsd:string	Yes	The sort criterion for the retrieval of elements. Default is random or server preferred sort.

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.1.1 Example (Informative)

C.1.1.1 Request

```
POST /exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/search HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

maxEntries=10&
searchKey=*my*&
resourceURL=http%3A%2F%2FexampleAPI%2Fucd%2Fv1%2Ftel%3A%2B19585550100%2Fmyfolder&
sortCriterion=Ascending by Date
```

C.1.1.2 Response

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 07:51:50 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ucd:searchResult xmlns:ucd="urn:oma:xml:rest:netapi:ucd:1">
 <result>
  <folderSearchResult>
   <folder>
    <folderAttributes>
     <root>false</root>
     <size>15649778</size>
     <createTime>2014-01-05T06:03:05Z</createTime>
     <filesNumber>3</filesNumber>
     <subFoldersNumber>0</subFoldersNumber>
     <owner>George Smith</owner>
     <accessControlList>
      <accessControlEntry>
       <acetype>ALLOW</acetype>
       <identifier>OWNER@</identifier>
       <aceflags>DIRECTORY_INHERIT</aceflags>
       <acemask>LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL, WRITE_OWNER,
```

```
SYNCHRONIZE </acemask>
      </accessControlEntry>
     </accessControlList>
    </folderAttributes>
    <files>
     <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel001.pdf</resourceURL>
     </reference>
     <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel002.pdf</resourceURL>
     </reference>
     <reference>

<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel003.pdf</resourceURL>
     </reference>
    </files>
    <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel</resourceURL>
   </folder>
  </folderSearchResult>
  <fileSearchResult>
   <file>
    <fileAttributes>
     <fileType>doc</fileType>
     <size>246918</size>
     <createTime>2014-01-09T18:20:40Z</createTime>
     <owner>George Smith</owner>
     <hash>
       <algorithm>sha-1</algorithm>
       <value>86D471913EE4B1DF2F58231FE8653BBCF371362F</value>
     </hash>
     <share>
      <isShare>false</isShare>
     </share>
    </fileAttributes>
    <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc</resourceURL>
   </file>
   <file>
    <fileAttributes>
     <fileType>jpg</fileType>
     <size>549778</size>
```

```
    <createTime>2014-01-19T08:30:50Z</createTime>
    <owner>George Smith</owner>
    <hash>
      <algorithm>sha-1</algorithm>
      <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
    </hash>
    <share>
     <isShare>true</isShare>

      <shareLink rel="FileShareLink" href="http://example.com/GeorgeSmith/mydocument2.jpg"/>
     <accessCode>12345</accessCode>
    </share>
    <revisionList>

<reference><resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/%myfolder%2Fmydocument/mydocument2.j
pg.rev001</resourceURL>
       </reference>
     </revisionList>
   </fileAttributes>
   <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</resourceURL>
  </file>
 </fileSearchResult>
</result>
 <resourceURL>http://exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/search</resourceURL>
</ucd:searchResult>
```

Appendix D. JSON examples (Informative)

JSON (JavaScript Object Notation) is a Light-weight, text-based, language-independent data interchange format. It provides a simple means to represent basic name-value pairs, arrays and objects. JSON is relatively trivial to parse and evaluate using standard JavaScript libraries, and hence is suited for REST invocations from browsers or other processors with JavaScript engines. Further information on JSON can be found at [RFC 4627].

The following examples show the request and response for various operations using the JSON data format. The examples follow the XML to JSON serialization rules in [REST_NetAPI_Common]. A JSON response can be obtained by using the content type negotiation mechanism specified in [REST_NetAPI_Common].

For full details on the operations themselves please refer to the section number indicated.

D.1 Retrieve information about a folder (section 6.1.3.1)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"folder": {
   "files": {"reference": [
      {"resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc"},
      {"resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg"}
   ]},
   "folderAttributes": {
      "accessControlList": {"accessControlEntry": {
         "aceflags": "DIRECTORY_INHERIT",
         "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
         "acetype": "ALLOW",
         "identifier": "OWNER@"
      }},
      "createTime": "2014-01-19T08:30:50Z",
      "filesNumber": "2",
      "owner": "George Smith",
      "root": "false",
      "size": "5030248",
      "subFoldersNumber": "1"
   },
   "resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument",
   "subFolders": {"reference": {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel"}}
}}
```

D.2 Retrieve information about a non-existent folder (section 6.1.3.2)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fotherdocument  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 404 Not Found
Content-Type: application/json
Content-Length: nnnn
Date: Fri, 17 Jan 2014 17:51:59 GMT

{"requestError": {
   "link": {
      "href": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fotherdocument",
      "rel": "folder"
   },
   "serviceException": {
      "messageId": "SVC0004",
      "text": "No valid addresses provided in message part %1",
      "variables": "Request-URI"
   }
}}
```

D.3 Retrieve information about a large folder (section 6.1.3.3)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument?maxEntries=1  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"folder": {
   "cursor": "cursor111",
   "folderAttributes": {
      "accessControlList": {"accessControlEntry": {
         "aceflags": "DIRECTORY_INHERIT",
         "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
         "acetype": "ALLOW",
```

```
            "identifier": "OWNER@"
        }},
        "createTime": "2014-01-19T08:30:50Z ",
        "filesNumber": "2",
        "owner": "George Smith",
        "root": "false",
        "size": "5030248",
        "subFoldersNumber": "1"
    },
    "resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument",
    "subFolders": {"reference": {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel"}}
}}
```


D.4 Retrieve information about a large folder (section 6.1.3.4)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument?fromCursor=abcdef%3Fcur%38194&maxEntries=2
HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"folder": {
    "files": {"reference": [
        {"resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc"},
        {"resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg"}
    ]},
    "folderAttributes": {
        "accessControlList": {"accessControlEntry": {
            "aceflags": "DIRECTORY_INHERIT",
            "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
            "acetype": "ALLOW",
            "identifier": "OWNER@"
        }},
        "createTime": "2014-01-19T08:30:50Z ",
        "filesNumber": "2",
        "owner": "George Smith",
        "root": "false",
        "size": "5030248",
        "subFoldersNumber": "1"
    },
    "resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument"
}}
```

D.5 Create folder, response with a location of created resource (section 6.1.4.1)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument HTTP/1.1
Accept: application/json
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
```

Response:

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument
Content-Type: application/json
Content-Length: nnnn

{"resourceReference": {"resourceURL": " http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument"}}
```

D.6 Create folder, response with a copy of created resource (section 6.1.4.2)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument HTTP/1.1
Accept: application/json
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

Response:

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument
Content-Type: application/json
Content-Length: nnnn

{"folder": {
   "folderAttributes": {
      "accessControlList": {"accessControlEntry": {
         "aceflags": "DIRECTORY_INHERIT",
         "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
         "acetype": "ALLOW",
         "identifier": "OWNER@"
      }},
      "createTime": "2014-01-19T08:30:50Z",
      "filesNumber": "0",
      "owner": "George Smith",
      "root": "false",
```

```
      "size": "0",
      "subFoldersNumber": "0"
   },
   "resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument"
}}
```


D.7 Delete a folder to recycle bin, response with “204 No Content” (section 6.1.6.1)

Request:

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument  HTTP/1.1
Host: example.com
Accept: application/json

{"deleteMode": {"deleteMode":DeleteToRecycleBin}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

D.8 Delete a folder permanently, response with “204 No Content” (section 6.1.6.2)

Request:

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument  HTTP/1.1
Host: example.com
Accept: application/json

{"deleteMode": {"deleteMode":DeletePermanently}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

D.9 Retrieve a folder’s attributes (section 6.2.3.1)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/folderAttributes
HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
```



```
Date: Thu, 04 Jun 2012 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"folderAttributes": {
   "accessControlList": {"accessControlEntry": {
      "aceflags": "DIRECTORY_INHERIT",
      "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
      "acetype": "ALLOW",
      "identifier": "OWNER@"
   }},
   "createTime": "2014-01-19T08:30:50Z",
   "filesNumber": "2",
   "owner": "George Smith",
   "root": "false",
   "size": "5030248",
   "subFoldersNumber": "1"
}}
```

D.10 Update individual folder ACL information (section 6.2.4.1)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/folderAttributes/acl HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
Content-Type: application/json
Content-Length: nnnn

{"accessControlList": {"accessControlEntry": {
   "aceflags": "DIRECTORY_INHERIT",
   "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD",
   "acetype": "ALLOW",
   "identifier": "OWNER@"
}}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"accessControlList": {"accessControlEntry": {
   "aceflags": "DIRECTORY_INHERIT",
   "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD",
   "acetype": "ALLOW",
   "identifier": "OWNER@"
}}}
```

D.11 Downloading a file (section 6.3.3.1)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Length: nnnn

[mydocument1.doc's content]
```

D.12 Uploading a file, response with a location of created resource (section 6.3.4.1)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg HTTP/1.1
Accept: application/json
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
Content-Type: image/jpg
Content-Length: nnnn
MIME-Version: 1.0

[mydocument2.jpg's content]
```

Response:

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg
Content-Type: application/json
Content-Length: nnnn


{"resourceReference": {"resourceURL": "
http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg"}}
```

D.13 Uploading a file, response with a copy of created resource (section 6.3.4.2)

Request:


```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg HTTP/1.1
Accept: application/json
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
Content-Type: image/jpg
Content-Length: nnnn
MIME-Version: 1.0

[mydocument2.jpg's content]
```

Response:

```
HTTP/1.1 201 Created
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg
Content-Type: application/json
Content-Length: nnnn

{"file": {
   "fileAttributes": {
      "accessControlList": {"accessControlEntry": {
         "aceflags": "FILE_INHERIT",
         "acemask": "READ_DATA, WRITE_DATA, APPEND_DATA, READ_NAMED_ATTRS, WRITE_NAMED_ATTRS,
READ_ATTRIBUTES, WRITE_ATTRIBUTES, DELETE, READ_ACL, WRITE_ACL, WRITE_OWNER, SYNCHRONIZE",
         "acetype": "ALLOW",
         "identifier": "OWNER@"
      }},
      "createTime": "2014-01-19T08:30:50Z",
      "fileType": "jpg",
      "hash": {
         "algorithm": "sha-1",
         "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
      },
      "metadataList": {"metadata": [
         {
            "name": "publisher",
            "value": "XYZ"
         },
         {
            "name": "project",
            "value": "abc"
         },
         {
            "name": "department",
            "value": "Sales"
         }
      ]},
      "owner": "George Smith",
      "revisionList": {"reference": {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg.rev001"}},
      "share": {"isShare": "false"},
      "size": "598208"
   },
   "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg"
}}
```

D.14 Updating file in range, response with a copy of created resource (section 6.3.4.3)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc HTTP/1.1
Accept: application/json
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
Content-Type: binary/oct-stream
Range:bytes=start-end

[mydocument1.doc's updating content]
```

Response:

```
HTTP/1.1 200 OK
Date: Tue, 19 Jan 201408:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc
Content-Type: application/json
Content-Range: start-end/size

{"file": {
   "fileAttributes": {
      "accessControlList": {"accessControlEntry": {
         "aceflags": "DIRECTORY_INHERIT",
         "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
         "acetype": "ALLOW",
         "identifier": "OWNER@"
      }},
      "accessTime": "2014-01-10T12:10:30Z",
      "createTime": "2014-01-09T18:20:40Z",
      "fileType": "doc",
      "hash": {
         "algorithm": "sha-1",
         "value": "86D471913EE4B1DF2F58231FE8653BBCF371362F"
      },
      "metadataList": {"metadata": [
         {
            "name": "publisher",
            "value": "HZ"
         },
         {
            "name": "project",
            "value": "abc"
         },
         {
            "name": "department",
            "value": "Sales"
         }
      ]},
      "modifyTime": "2014-01-19T08:30:50Z",
      "owner": "George Smith",
      "share": {"isShare": "false"},
```

```
      "size": "230208"
   },
   "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc"
}}
```

D.15 Delete a file to recycle bin, response with “204 No Content” (section 6.3.6.1)

Request:

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc  HTTP/1.1
Host: example.com
Accept: application/json

{"deleteMode": {"deleteMode":DeleteToRecycleBin}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

D.16 Delete a file to permanently, response with “204 No Content” (section 6.3.6.2)

Request:

```
DELETE /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc  HTTP/1.1
Host: example.com
Accept: application/json

{"deleteMode": {"deleteMode":DeletePermanently}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 05 Sep 2013 06:05:09 GMT
```

D.17 Retrieve a file’s attributes (section 6.4.3.1)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes
HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2012 02:51:59 GMT
```



```
Content-Type: application/json
Content-Length: nnnn

{"fileAttributes": {
   "accessControlList": {"accessControlEntry": {
      "aceflags": "DIRECTORY_INHERIT",
      "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
      "acetype": "ALLOW",
      "identifier": "OWNER@"
   }},
   "createTime": "2014-01-19T08:30:50Z",
   "fileType": "jpg",
   "hash": {
      "algorithm": "sha-1",
      "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
   },
   "metadataList": {"metadata": [
      {
         "name": "publisher",
         "value": "XYZ"
      },
      {
         "name": "project",
         "value": "abc"
      },
      {
         "name": "department",
         "value": "Sales"
      }
   ]},
   "owner": "George Smith",
   "revisionList": {"reference": {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg.rev001"}},
   "share": {"isShare": "false"},
   "size": "598208"
}}
```

D.18 Update individual file's attribute of fileType (section 6.4.4.1)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/fileType HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
Content-Type: application/json
Content-Length: nnnn

{"fileType": "png"}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"fileType": "png"}
```

D.19 Update individual file's attribute of metadata (section 6.4.4.2)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/metadata HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
Content-Type: application/json
Content-Length: nnnn

{"metadataList": {"metadata": [
   {
      "name": "publisher",
      "value": "XYZ"
   },
   {
      "name": "project",
      "value": "def"
   },
   {
      "name": "department",
      "value": "Sales"
   }
]}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"metadataList": {"metadata": [
   {
      "name": "publisher",
      "value": "XYZ"
   },
   {
      "name": "project",
      "value": "def"
   },
   {
```

```
      "name": "department",
      "value": "Sales"
   }
]}}
```

D.20 Update individual file ACL information (section 6.4.4.3)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/acl HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
Content-Type: application/json
Content-Length: nnnn

{"accessControlList": {"accessControlEntry": {
   "aceflags": "FILE_INHERIT",
   "acemask": "READ_DATA, WRITE_DATA, APPEND_DATA, DELETE, READ_ACL, WRITE_ACL, WRITE_OWNER, SYNCHRONIZE",
   "acetype": "ALLOW",
   "identifier": "OWNER@"
}}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"accessControlList": {"accessControlEntry": {
   "aceflags": "FILE_INHERIT",
   "acemask": "READ_DATA, WRITE_DATA, APPEND_DATA, DELETE, READ_ACL, WRITE_ACL, WRITE_OWNER, SYNCHRONIZE",
   "acetype": "ALLOW",
   "identifier": "OWNER@"
}}}
```

D.21 Update individual file sharing information (section 6.4.4.4)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg/fileAttributes/share HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
Content-Type: application/json
Content-Length: nnnn

{"share": {
   "accessCode": "12345",
```

```
   "isShare": "true",
   "sharelink": {
      "href": "http://example.com/GeorgeSmith/mydocument2.jpg",
      "rel": "FileShareLink"
   }}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Mar 2014 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"share": {
   "accessCode": "12345",
   "isShare": "true",
   "shareLink": "http://example.com/GeorgeSmith/mydocument2.jpg"
}}
```

D.22 List recycle bin (section 6.5.3.1)

Request:

```
GET /exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin  HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 09:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"recycleBin": {
   "recycleBinItemList": {"recycleBinItem": [
      {
         "name": "mypicture",
         "originalPath": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmypicture",
         "recycleBinItemAttributes": {
            "createTime": "2014-01-19T08:30:50Z",
            "deleteTime": "2014-02-01T07:29:45Z",
            "size": "51198208"
         },
         "type": "0"
      },
      {
         "name": "novel111.pdf",
         "originalPath":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel111.pdf",
```

```
            "recycleBinItemAttributes": {
               "createTime": "2014-01-01T08:30:50Z",
               "deleteTime": "2014-01-03T07:29:45Z",
               "fileType": "pdf",
               "size": "2030248"
            },
            "type": "1"
         }
   ]},
   "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin"
}}
```

D.23 Revoking recycle bin items (section 6.5.4.1)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin HTTP/1.1
Host: example.com
Accept: application/json

{"recycleBin": {
   "recycleBinItemList": {
      "recycleBinItem": {
         "name": "novel111.pdf",
         "originalPath":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel111.pdf",
         "recycleBinItemAttributes": {
            "createTime": "2014-01-01T08:30:50Z",
            "deleteTime": "2014-01-03T07:29:45Z",
            "fileType": "pdf",
            "size": "2030248"
         },
         "type": "1"
      },
      "recycleBinTreatment": "Revoke"
   },
   "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 05 May 2014 06:05:09 GMT
```

D.24 Clean all items in recycle bin (section 6.5.4.2)

Request:

```
PUT /exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin HTTP/1.1
Host: example.com
Accept: application/json

{"recycleBin": {
   "recycleBinItemList": {"recycleBinTreatment": "Delete"},
```

```
   "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/recyclebin"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 05 May 2014 06:05:09 GMT
```

D.25 Search (section 6.6.5.1)

Request:

```
POST /exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/search HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Accept: application/xml
Content-Type: application/json
Content-Length: nnnn

{"searchCriteria": {
   "maxEntries": "10",
   "searchKey": "*my*",

    "searchScope": {"resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder"},
   "sortCriterion": "Ascending by Date"
}}
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 07:51:50 GMT
Content-Type: application/json
Content-Length: nnnn

{"searchResult": {
   "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/search",
   "result": {
      "fileSearchResult": {"file": [
         {
            "fileAttributes": {
               "createTime": "2014-01-09T18:20:40Z",
               "fileType": "doc",
               "hash": {
                  "algorithm": "sha-1",
                  "value": "86D471913EE4B1DF2F58231FE8653BBCF371362F"
               },
               "owner": "George Smith",
               "share": {"isShare": "false"},
               "size": "246918"
            },
            "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument1.doc"
```

```
        },
        {
           "fileAttributes": {
              "createTime": "2014-01-19T08:30:50Z",
              "fileType": "jpg",
              "hash": {
                 "algorithm": "sha-1",
                 "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
              },
              "owner": "George Smith",
              "revisionList": {"reference": {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg.rev001"}},
              "share": {
                 "accessCode": "12345",
                 "isShare": "true",
                 "sharelink": {
                   "href": "http://example.com/GeorgeSmith/mydocument2.jpg",
                   "rel": "FileShareLink" }},
              "size": "549778"
           },
           "resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg"
        }
     ]},
     "folderSearchResult": {"folder": {
        "files": {"reference": [
           {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel001.pdf"},
           {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel002.pdf"},
           {"resourceURL":
"http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel003.pdf"}
        ]},
        "folderAttributes": {
           "accessControlList": {"accessControlEntry": {
              "aceflags": "DIRECTORY_INHERIT",
              "acemask": "LIST_DIRECTORY, ADD_FILE, ADD_SUBDIRECTORY, DELETE_CHILD, READ_ACL, WRITE_ACL,
WRITE_OWNER, SYNCHRONIZE ",
              "acetype": "ALLOW",
              "identifier": "OWNER@"
           }},
           "createTime": "2014-01-05T06:03:05Z",
           "filesNumber": "3",
           "owner": "George Smith",
           "root": "false",
           "size": "15649778",
           "subFoldersNumber": "0"
        },
        "resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel"
     }}
  }
}}
```


D.26 List the shared files (section 6.7.5.1)

Request:


```
POST /exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/listShare HTTP/1.1
Accept: application/json
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 14 Mar 2014 07:51:50 GMT
Content-Type: application/json
Content-Length: nnnn

{"ShareList": {"share": {
   "accessCode": "12345",
   "isShare": "true",
   "sharelink": {
      "href": "http://example.com/GeorgeSmith/mydocument2.jpg",
      "rel": "FileShareLink" }
}}}
```

Appendix E. Operations mapping to a pre-existing baseline specification (Informative)

As this specification does not have a baseline specification, this appendix is empty.

Appendix F. Light-weight Resources (Informative)

The following table lists all UCD data structure elements that can be accessed individually as Light-weight Resources.

For each Light-weight Resource, the following information is provided: corresponding root element name, root element type and [ResourceRelPath] string.

Type of Light-weight Resources (and references to data structures)	Element/attribute that can be accessed as Light-weight Resource	Root element name for the Light-weight Resource	Root element type for the Light-weight Resource	[ResourceRelPath] string that needs to be appended to the corresponding Heavy-weight Resource URL
File (5.2.2.1)	fileAttributes	fileAttributes	FileAttributes	fileAttributes
FileAttributes (5.2.2.2)	fileType	fileType	xsd:string	fileAttributes/fileType
	metadataList	metadataList	MetadataList	fileAttributes/metadata
	accessControlList	accessControlList	AccessControlList	fileAttributes/acl
	share	share	Share	fileAttributes/share
Folder (5.2.2.8)	folderAttributes	folderAttributes	FolderAttributes	folderAttributes
FolderAttributes (5.2.2.9)	accessControlList	accessControlList	AccessControlList	folderAttributes/acl

Appendix G. Authorization aspects (Normative)

This appendix specifies how to use the RESTful Unified Cloud Disk API in combination with some authorization frameworks.

G.1 Use with OMA Authorization Framework for Network APIs

The RESTful UCD API MAY support the authorization framework defined in [Autho4API_10].

A RESTful UCD API supporting [Autho4API_10]:

- SHALL conform to section D.1 of [REST_NetAPI_Common];
- SHALL conform to this section G.1.

G.1.1 Scope values

G.1.1.1 Definitions

In compliance with [Autho4API_10], an authorization server serving clients requests for getting authorized access to the resources exposed by the RESTful UCD API:

- SHALL support the scope values defined in the table below;
- MAY support scope values not defined in this specification.

Scope value	Description	For one-time access token
oma_rest_ucd.all_{apiVersion}	Provide access to all defined operations on the resources in this version of the API. The {apiVersion} part of this identifier SHALL have the same value as the “apiVersion” URL variable which is defined in section 5.1. This scope value is the union of the other scope values listed in next rows of this table.	No
oma_rest_ucd.file	Provide access to all defined operations(e.g., read, delete, update, etc.) on file	No
oma_rest_ucd.folder	Provide access to all defined operations(e.g., read, delete, etc.) on folder	No
oma_rest_ucd.recycleBin	Provide access to all defined operations(e.g., read, delete, etc.) on recycleBin	No
oma_rest_ucd.search	Provide access to all defined operations(e.g., allowed or not allowed to search) on search	No
oma_rest_ucd.listShare	Provide access to all defined operations on listShare	No

Table 1: Scope values for RESTful UCD API

G.1.1.2 Downscoping

In the case where the client requests authorization for "oma_rest_ucd.all_{apiVersion}" scope, the authorization server and/or resource owner MAY restrict the granted scope to some of the following scope values:

- oma_rest_ucd.file
- oma_rest_ucd.folder
- oma_rest_ucd.recyclebin
- oma_rest_ucd.search
- oma_rest_ucd.listShare

G.1.1.3 Mapping with resources and methods

Tables in this section specify how the scope values defined in section G.1.1.1 for the RESTful UCD API map to the REST resources and methods of this API. In these tables, the root "oma_rest_ucd." of scope values is omitted for readability reasons.

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
A file	/{folderName}/{fileName}	6.3	**all_{apiVersion}** or **file**	**all_{apiVersion}** or **file**	n/a	**all_{apiVersion}** or **file**
Individual File Attributes	/{folderName}/{fileName}/[ResourceRelPath]	6.4	**all_{apiVersion}** or **file**	**all_{apiVersion}** or **file**	n/a	n/a

Table 2: Required scope values for: managing files

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
A folder	/{folderName}	6.1	**all_{apiVersion}** or **folder**	**all_{apiVersion}** or **folder**	n/a	**all_{apiVersion}** or **folder**
Individual folder attributes	/{folderName}/[ResourceRelPath]	6.2	**all_{apiVersion}** or **folder**	**all_{apiVersion}** or **folder**	n/a	n/a

Table 3: Required scope values for: managing folder

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Recycle Bin	/recycleBin	6.5	**all_{apiVersion}** or **recycleBin**	**all_{apiVersion}** or **recycleBin**	n/a	n/a

Table 4: Required scope values for: managing recyclebin

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
File or Folder search	/operations/search	6.6	n/a	n/a	**all_{apiVersion}** or **search**	n/a

Table 5: Required scope values for: managing search

Resource	URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
List shared files	/operations/listShare	6.7	n/a	n/a	**all_{apiVersion}** or **listShare**	n/a

Table 6: Required scope values for: managing listShare

G.1.2 Use of ‘acr:auth’

This section specifies the use of ‘acr:auth’ in place of an end user identifier in a resource URL path.

An ‘acr’ URI of the form ‘acr:auth’, where ‘auth’ is a reserved keyword MAY be used to avoid exposing a real end user identifier in the resource URL path.

A client MAY use ‘acr:auth’ in a resource URL in place of a {userId} when the RESTful UCD API is used in combination with [Autho4API_10].

In the case the RESTful UCD API supports [Autho4API_10], the server:

- SHALL accept ‘acr:auth’ as a valid value for the resource URL variable {userId}
- SHALL conform to [REST_NetAPI_Common] section 5.8.1.1 regarding the processing of ‘acr:auth’.