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
6.3 Resource: A File ................................................................................................................................. 40

6.3.1 Request URL variables .................................................................................................................. 40
6.3.2 Response Codes and Error Handling .............................................................................................. 40
6.3.3 GET ................................................................................................................................................ 41
6.3.4 PUT .............................................. 41

6.3.4.1 Example 1: Uploading a file, response with a location of created resource (Informative) .......... 41
6.3.4.1.1 Request .............................................................................................................................. 41
6.3.4.1.2 Response ........................................................................................................................ 41
6.3.4.2 Example 2: Uploading a file, response with a copy of created resource (Informative) ............ 42
6.3.4.2.1 Request ............................................................................................................................ 42
6.3.4.2.2 Response ........................................................................................................................ 42
6.3.4.3 Example 3: Updating file in range, response with a copy of created resource (Informative) .... 43
6.3.4.3.1 Request ............................................................................................................................ 43
6.3.4.3.2 Response ........................................................................................................................ 43
6.3.5 POST ............................................................................................................................................. 44
6.3.6 DELETE ......................................................................................................................................... 44

6.3.6.1 Example 1: Delete a file to recycle bin, response with “204 No Content” (Informative) .......... 44
6.3.6.1.1 Request ............................................................................................................................ 44
6.3.6.1.2 Response ........................................................................................................................ 45
6.3.6.2 Example 2: Delete a file permanently, response with “204 No Content” (Informative) ......... 45
6.3.6.2.1 Request ............................................................................................................................ 45
6.3.6.2.2 Response ........................................................................................................................ 45

6.4 Resource: Individual File Attributes .............................................................................................. 45

6.4.1 Request URL variables .................................................................................................................. 45
6.4.2 Response Codes and Error Handling .............................................................................................. 46
6.4.3 GET .............................................................................................................................................. 46

6.4.3.1 Example: Retrieve a file’s attributes (Informative) .................................................................... 46
6.4.3.1.1 Request ............................................................................................................................ 46
6.4.3.1.2 Response ........................................................................................................................ 46
6.4.4 PUT .............................................................................................................................................. 47
6.4.5 POST ........................................................................................................................................... 50
6.4.6 DELETE ....................................................................................................................................... 50

6.5 Resource: RecycleBin ....................................................................................................................... 50
6.5.1 Request URL variables ................................................................. 50
6.5.2 Response Codes and Error Handling ........................................ 51
6.5.3 GET .......................................................................................... 51
6.5.4 PUT .......................................................................................... 52
   6.5.4.1 Example 1: Revoking recycle bin items (Informative) ......... 52
       6.5.4.1.1 Request ........................................................................... 52
       6.5.4.1.2 Response ................................................................. 52
6.5.4.2 Example 2: Clean the recycle bin (Informative) .................. 53
       6.5.4.2.1 Request ........................................................................... 53
       6.5.4.2.2 Response ................................................................. 53
6.5.5 POST ....................................................................................... 53
6.5.6 DELETE .................................................................................. 53

6.6 RESOURCE: SEARCH FOR FILES OR FOLDERS .......................... 53
   6.6.1 Request URL variables ............................................................ 53
   6.6.2 Response Codes and Error Handling ....................................... 54
   6.6.3 GET ....................................................................................... 54
   6.6.4 PUT ....................................................................................... 54
   6.6.5 POST ..................................................................................... 54
   6.6.6 DELETE ............................................................................... 56

6.7 RESOURCE: LIST THE SHARED FILES ...................................... 56
   6.7.1 Request URL variables ............................................................ 57
   6.7.2 Response Codes and Error Handling ....................................... 57
   6.7.3 GET ....................................................................................... 57
   6.7.4 PUT ....................................................................................... 57
   6.7.5 POST ..................................................................................... 57
   6.7.6 DELETE ............................................................................... 58

7. FAULT DEFINITIONS ...................................................................... 59

7.1 SERVICE EXCEPTIONS ............................................................... 59
7.2 POLICY EXCEPTIONS .................................................................. 59

APPENDIX A. CHANGE HISTORY (INFORMATIVE) ............................. 60
   A.1 APPROVED VERSION HISTORY ................................................ 60
   A.2 DRAFT/CANDIDATE VERSION 1.0 HISTORY ................................. 60

APPENDIX B. STATIC CONFORMANCE REQUIREMENTS (NORMATIVE) .............................................................................. 61
   B.1 SCR FOR REST.UCD SERVER ................................................... 61
       B.1.1 SCR for REST.UCD.Folder Server ...................................... 61
       B.1.2 SCR for REST.UCD.Folder.Attr Server ......................... 61
       B.1.3 SCR for REST.UCD.File Server ....................................... 61
       B.1.4 SCR for REST.UCD.File.Attr Server ............................ 61
       B.1.5 SCR for REST.UCD.Recyclebin Server ....................... 62
       B.1.6 SCR for REST.UCD.Search Server .............................. 62
       B.1.7 SCR for REST.UCD.ListShare Server ...................... 62

APPENDIX C. APPLICATION/X-WWW-FORM-URLENCODED REQUEST FORMAT FOR POST OPERATIONS (NORMATIVE) ................................................................. 63
   C.1 SEARCH FOLDERS OR FILES ................................................ 63
       C.1.1 Example (Informative) .......................................................... 64
           C.1.1.1 Request ........................................................................ 64
           C.1.1.2 Response .................................................................... 64

APPENDIX D. JSON EXAMPLES (INFORMATIVE) .............................. 67
   D.1 RETRIEVE INFORMATION ABOUT A FOLDER (SECTION 6.1.3.1) .................. 67
   D.2 RETRIEVE INFORMATION ABOUT A NON-EXISTENT FOLDER (SECTION 6.1.3.2) .......... 68
   D.3 RETRIEVE INFORMATION ABOUT A LARGE FOLDER (SECTION 6.1.3.3) ................... 68
   D.4 RETRIEVE INFORMATION ABOUT A LARGE FOLDER (SECTION 6.1.3.4) ................... 69
   D.5 CREATE FOLDER, RESPONSE WITH A LOCATION OF CREATED RESOURCE (SECTION 6.1.4.1) .... 70
   D.6 CREATE FOLDER, RESPONSE WITH A COPY OF CREATED RESOURCE (SECTION 6.1.4.2) .......... 70
   D.7 DELETE A FOLDER TO RECYCLE BIN, RESPONSE WITH “204 NO CONTENT” (SECTION 6.1.6.1) .......... 71

© 2014 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.
APPENDIX F. OPERATIONS MAPPING TO A PRE-EXISTING BASELINE SPECIFICATION (INFORMATIVE) .......................................................... 85

APPENDIX G. AUTHORIZATION ASPECTS (NORMATIVE) .......................................................................................................................... 87

G.1 USE WITH OMA AUTHORIZATION FRAMEWORK FOR NETWORK APIs ......................................................................................... 87

G.1.1 Scope values .................................................................................................................................................................................. 87

G.1.1.1 Definitions ............................................................................................................................................................................... 87

G.1.1.2 Downscoping .......................................................................................................................................................................... 88

G.1.1.3 Mapping with resources and methods ...................................................................................................................................... 88

G.1.2 Use of ‘acr:auth’ ............................................................................................................................................................................ 91

Figures

Figure 1 UCD-2 Resource structure defined by this specification .................................................................................................. 13

Figure 2 Folder Operations .................................................................................................................................................... 26

Figure 3 Folder attributes operations ........................................................................................................................................ 27

Figure 4 File Operations ............................................................................................................................................................. 27

Figure 5 File attributes operations ........................................................................................................................................... 28

Figure 6 Recycle bin operations .................................................................................................................................................. 29

Figure 7 Search operation ............................................................................................................................................................ 29

Figure 8 List shared file operation ............................................................................................................................................. 30

Tables

Table 1: Scope values for RESTful UCD API ........................................................................................................................................ 87

Table 2: Required scope values for: managing files ......................................................................................................................... 89

Table 3: Required scope values for: managing folder ....................................................................................................................... 89

Table 4: Required scope values for: managing recyclebin .............................................................................................................. 89

© 2014 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document
Table 5: Required scope values for: managing search .........................................................................................................90
Table 6: Required scope values for: managing listShare .....................................................................................................90
1. Scope

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

2.1 Normative References


[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/


[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-urlencoded (Appendix C) is supported]


2.2 Informative References


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.

<table>
<thead>
<tr>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Heavy-weight Resource</strong></td>
</tr>
<tr>
<td>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.</td>
</tr>
<tr>
<td><strong>Light-weight Resource</strong></td>
</tr>
<tr>
<td>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.</td>
</tr>
</tbody>
</table>

3.3 Abbreviations

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

Figure 1 UCD-2 Resource structure defined by this specification
Purpose: To allow client to manage folder or file

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>A folder</td>
<td>/{folderName}</td>
<td>Folder (used for GET/response, PUT/response)</td>
<td>List folder information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>common:ResourceReference (optional alternative for PUT response)</td>
<td>Create folder</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DeleteMode (used for DELETE request)</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: folderName is absolute value including path</td>
<td>Delete folder to recycle bin or permanently</td>
</tr>
<tr>
<td>Individual folder</td>
<td>/{folderName}/{ResourceRelPath}</td>
<td>The data structure corresponds to an element within the Folder structure pointed out by the resource URL. (used for PUT/GET)</td>
<td>Retrieve individual folder attributes</td>
</tr>
<tr>
<td>attributes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A file</td>
<td>/{folderName}/{fileName}</td>
<td>File (used for PUT/response)</td>
<td>Download file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>common:ResourceReference (optional alternative for PUT response)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DeleteMode (used for DELETE request)</td>
<td></td>
</tr>
<tr>
<td>Resource</td>
<td>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</td>
<td>Data Structures</td>
<td>HTTP verbs</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Individual file attributes</td>
<td>{folderName}/{fileName}/{ResourceRelPath}</td>
<td>The data structure corresponds to an element within the File structure pointed out by the resource URL. (used for PUT/GET)</td>
<td>Retrieve individual file attributes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Update individual file attributes including fileType, metadata, ACL information, sharing information</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>
### Purpose: To allow client to manage Recycle bin

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recycle bin</td>
<td>/recycleBin</td>
<td>RecycleBin (used for GET response, PUT request)</td>
<td>List recycle bin</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search for file or folder</td>
<td>/operations/search</td>
<td>SearchCriteria (used for POST request)</td>
<td>No</td>
</tr>
<tr>
<td>List of shared files</td>
<td>/operations/listShare</td>
<td>ShareList (used for POST response)</td>
<td>No</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>[ResourceRelPath]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileAttributes</td>
<td>FileAttributes</td>
<td>Yes</td>
<td>fileAttributes</td>
<td>List of attributes associated with the file</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Not applicable</td>
<td>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.</td>
</tr>
</tbody>
</table>

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.

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

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.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>Metadata [0..unbounded]</td>
<td>Yes</td>
<td>A list of metadata.</td>
</tr>
</tbody>
</table>

### 5.2.2.4 Type: Metadata

The user self define metadata.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>xsd:string</td>
<td>No</td>
<td>Metadata name.</td>
</tr>
<tr>
<td>value</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Metadata value.</td>
</tr>
</tbody>
</table>

### 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).

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessControlEntry</td>
<td>AccessControlEntry [0..unbounded]</td>
<td>Yes</td>
<td>A list of resource references</td>
</tr>
</tbody>
</table>

### 5.2.2.6 Type: AccessControlEntry

The access control entry information.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acetype</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The access control entry types. See [RFC3530]</td>
</tr>
<tr>
<td>identifier</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The user identifier, special &quot;who&quot; see [RFC3530]</td>
</tr>
<tr>
<td>aceflags</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The semantics of the ACE. See [RFC3530]</td>
</tr>
<tr>
<td>acemask</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The operations on a file or folder(directory in NFS v4) See [RFC3530]</td>
</tr>
</tbody>
</table>
### 5.2.2.7 Type: HashInformation

This type represents the file hash information.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithm</td>
<td>xsd:string</td>
<td>No</td>
<td>The hash algorithm used (only &quot;sha-1&quot; currently supported). See [RFC5547].</td>
</tr>
<tr>
<td>value</td>
<td>xsd:hexBinary</td>
<td>No</td>
<td>The hash value of the file. See [RFC5547].</td>
</tr>
</tbody>
</table>

### 5.2.2.8 Type: Folder

Individual folder

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>[ResourceRelPath]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folderAttributes</td>
<td>FolderAttributes</td>
<td>Yes</td>
<td>folderAttributes</td>
<td>List of attributes associated with the folder</td>
</tr>
<tr>
<td>cursor</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Not applicable</td>
<td>If the lists of sub-folders and files are both complete, this element is omitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>subFolders</td>
<td>ReferenceList</td>
<td>Yes</td>
<td>Not applicable</td>
<td>List of sub-folders under this folder. The client SHALL NOT include this element in PUT requests.</td>
</tr>
<tr>
<td>files</td>
<td>ReferenceList</td>
<td>Yes</td>
<td>Not applicable</td>
<td>List of files under this folder. The client SHALL NOT include this element in PUT requests.</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Not applicable</td>
<td>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.</td>
</tr>
</tbody>
</table>

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.

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

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].
5.2.2.10  **Type: ReferenceList**

List of object references

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reference</td>
<td>common:ResourceReference [0..unbounded]</td>
<td>Yes</td>
<td>A list of resource references</td>
</tr>
</tbody>
</table>

5.2.2.11  **Type: DeleteMode**

The delete mode

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>deleteMode</td>
<td>DeleteModeEnum</td>
<td>No</td>
<td>The delete mode</td>
</tr>
</tbody>
</table>

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

5.2.2.12  **Type: Share**

The file sharing.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isShare</td>
<td>xsd:boolean</td>
<td>No</td>
<td>Whether share file or not. Default is false.</td>
</tr>
<tr>
<td>shareLink</td>
<td>common: Link</td>
<td>Yes</td>
<td>The file share link</td>
</tr>
<tr>
<td>accessCode</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The code to access the file via the link.</td>
</tr>
</tbody>
</table>

5.2.2.13  **Type: ShareList**

The file sharing list.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>share</td>
<td>Share[0..unbounded]</td>
<td>Yes</td>
<td>List of file sharing</td>
</tr>
</tbody>
</table>

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

5.2.2.14  **Type: SearchCriteria**

Search criteria.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromCursor</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result</td>
<td>Yes</td>
<td>The search results. Number of results MAY be limited by the server.</td>
</tr>
</tbody>
</table>
| 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.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folderSearchResult</td>
<td>FolderList</td>
<td>Yes</td>
<td>List of folders.</td>
</tr>
<tr>
<td>fileSearchResult</td>
<td>FileList</td>
<td>Yes</td>
<td>List of files.</td>
</tr>
</tbody>
</table>

5.2.2.17 Type: FolderList

The list of folder information.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folder</td>
<td>Folder[0..unbounded]</td>
<td>Yes</td>
<td>A list of folder.</td>
</tr>
</tbody>
</table>

5.2.2.18 Type: FileList

The list of file information.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file</td>
<td>File[0..unbounded]</td>
<td>Yes</td>
<td>A list of file.</td>
</tr>
</tbody>
</table>

5.2.2.19 Type: RecycleBin

List of Recycle Bin.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recycleBinItemList</td>
<td>RecycleBinItemList</td>
<td>Yes</td>
<td>List of Recycle Bin</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Self referring URL</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recycleBinItem</td>
<td>RecycleBinItem[0..unbounded]</td>
<td>Yes</td>
<td>A list of metadata.</td>
</tr>
</tbody>
</table>

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

### 5.2.2.22 Type: RecycleBinItemAttributes

The Recycle Bin item attributes.

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

### 5.2.3 Enumerations

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

#### 5.2.3.1 Enumeration: DeleteModeEnum

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeletePermanently</td>
<td>Remove from server storage and no revoke</td>
</tr>
<tr>
<td>DeleteToRecycleBin</td>
<td>Temporarily move to Recycle Bin and can revoke</td>
</tr>
</tbody>
</table>

#### 5.2.3.2 Enumeration: RecycleBinTreatmentEnum

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revoke</td>
<td>Revoke the Recycle Bin items.</td>
</tr>
<tr>
<td>Delete</td>
<td>Delete the Recycle Bin items.</td>
</tr>
</tbody>
</table>

### 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
  \[\text{http://}\{\text{serverRoot}\}/\text{ucd}/\{\text{apiVersion}\}/\{\text{userId}\}/\{\text{folderName}\}\]

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
  \[\text{http://}\{\text{serverRoot}\}/\text{ucd}/\{\text{apiVersion}\}/\{\text{userId}\}/\{\text{folderName}\}/[\text{ResourceRelPath}]\]
  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
  \[\text{http://}\{\text{serverRoot}\}/\text{ucd}/\{\text{apiVersion}\}/\{\text{userId}\}/\{\text{folderName}\}/[\text{ResourceRelPath}]\]
  Where [ResourceRelPath] is a light-weight relative resource URL, and in this case it shall be replaced with “folderAttributes/acl”
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:
  
  \[
  \text{http://[serverRoot]/ucd/[apiVersion]/[userId]/[folderName]/[filename]}\]

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”

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:
  \[http://\{serverRoot\}/ucd/\{apiVersion\}/\{userId\}/\{recycleBin\}\]
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](image)

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
  
  \[http://\{serverRoot\}/ucd/\{apiVersion\}/\{userId\}/operations/listShare\]
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.

Figure 8 List shared file operation
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:

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

<table>
<thead>
<tr>
<th>Name</th>
<th>Type/Values</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromCursor</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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;</td>
</tr>
<tr>
<td>maxEntries</td>
<td>xsd:int</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Type/Values</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromCursor</td>
<td>xsd:string</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>maxEntries</td>
<td>xsd:int</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

**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

**Response**

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

```xml
<?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></ucd:folder>
```
<accessControlEntry>
</accessControlEntry>
</folderAttributes>
<subFolders>
<reference>
</reference>
</subFolders>
<files>
<reference>
</reference>
<reference>
<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</resourceURL>
</reference>
</files>
</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>
        <cursor>cursor111</cursor>
        <subFolders>
            <reference>
            </reference>
        </subFolders>
    </folderAttributes>
</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%3?cur%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>
    </folderAttributes>
</ucd:folder>
<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>
  </reference>
  <reference>
    <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument/mydocument2.jpg</resourceURL>
  </reference>
</files>
<resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument</resourceURL>
</ucd>

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%2B195855550100/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%2B195855550100/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>
        <resourceURL>http://example.com/exampleAPI/ucd/v1/tel%3A%2B195855550100/myfolder%2Fmydocument</resourceURL>
    </folderAttributes>
</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
<?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
<?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:
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI</td>
</tr>
<tr>
<td>apiVersion</td>
<td>Version of the API client wants to use. The value of this variable is defined in section 5.1</td>
</tr>
<tr>
<td>userId</td>
<td>Identifier of user.</td>
</tr>
<tr>
<td>folderName</td>
<td>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.</td>
</tr>
<tr>
<td>[ResourceRelPath]</td>
<td>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.</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Light-weight Resource type</th>
<th>Method supported</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual folder attributes</td>
<td>GET</td>
<td>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.</td>
</tr>
<tr>
<td>individual folder ACL information</td>
<td>PUT</td>
<td>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.</td>
</tr>
</tbody>
</table>

### 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</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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI</td>
</tr>
<tr>
<td>apiVersion</td>
<td>Version of the API client wants to use. The value of this variable is defined in section 5.1</td>
</tr>
<tr>
<td>userId</td>
<td>Identifier of user.</td>
</tr>
<tr>
<td>folderName</td>
<td>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.</td>
</tr>
<tr>
<td>fileName</td>
<td>The file name.</td>
</tr>
</tbody>
</table>

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 2014 08:30:50 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>
</ucd:file>
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/octet-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>...
    </metadataList>
  </fileAttributes>
</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/uri%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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI</td>
</tr>
<tr>
<td>apiVersion</td>
<td>Version of the API client wants to use. The value of this variable is defined in section 5.1</td>
</tr>
<tr>
<td>userId</td>
<td>Identifier of user.</td>
</tr>
<tr>
<td>folderName</td>
<td>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.</td>
</tr>
<tr>
<td>fileName</td>
<td>The file name.</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>Light-weight Resource type</th>
<th>Method supported</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual file attributes</td>
<td>GET</td>
<td>Enables retrieve individual folder attributes. See column [ResourceRelPath] for element &quot;folderAttributes&quot; in section 5.2.2.1 for possible values for the Light-weight relative resource path.</td>
</tr>
<tr>
<td>individual file attributes including filetype, metadata, ACL information, sharing information</td>
<td>PUT</td>
<td>Enables update file attributes including fileType, metadata, ACL information, sharing information. See column [ResourceRelPath] for element &quot;folderAttributes/acl&quot; in section 5.2.2.2 for possible values for the Light-weight relative resource path.</td>
</tr>
</tbody>
</table>

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>
</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
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

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

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>

<?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

© 2014 Open Mobile Alliance Ltd. All Rights Reserved. Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.
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>

6.4.4.2 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>Server base url: hostname+port+base path. Port and base path are OPTIONAL.</td>
</tr>
<tr>
<td></td>
<td>Example: example.com/exampleAPI</td>
</tr>
<tr>
<td>Name</td>
<td>Type/Values</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>fromCursor</td>
<td>xsd:string</td>
</tr>
<tr>
<td>maxEntries</td>
<td>xsd:int</td>
</tr>
</tbody>
</table>

6.5.3.1 Example 1: List recycle bin  

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"?>  
<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>  
</recycleBinItemList>  
</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"?>
<recycleBinItemList>
<recycleBinItem>
<type>1</type>
<name>novel111.pdf</name>
<originalPath>http://example.com/exampleAPI/ucd/v1/tel%3A%2B19585550100/myfolder%2Fmydocument%2Fnovel/novel111.pdf</originalPath>
<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>

6.5.4.1.2 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/v1/tel%3A%2B19585550100/recycleBin HTTP/1.1
Host: example.com
Accept: application/xml

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

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

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

```xml
POST /exampleAPI/ucd/v1/tel%3A%2B19585550100/operations/search HTTP/1.1
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5c903
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>
          </reference>
          <reference>
          </reference>
          <reference>
          </reference>
        </files>
      </folder>
    </folderSearchResult>
  </result>
  <fileSearchResult>
    <file>
      <fileAttributes>
        <fileType>doc</fileType>
        <size>246918</size>
        <createTime>2014-01-09T18:20:40Z</createTime>
      </fileAttributes>
    </file>
  </fileSearchResult>
</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:

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

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)

**Request**

```plaintext
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
```

**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"/></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

A.1 Approved Version History

<table>
<thead>
<tr>
<th>Reference</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n/a</td>
<td>n/a</td>
<td>No prior version</td>
</tr>
</tbody>
</table>

A.2 Draft/Candidate Version 1.0 History

<table>
<thead>
<tr>
<th>Document Identifier</th>
<th>Date</th>
<th>Sections</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft Versions:</td>
<td>05 may 2014</td>
<td>All</td>
<td>Initial baseline OMA-CD-UCD-2014-0027-INP_REST_NetAPI_UCD_TS_base</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>All</td>
<td>OMA-CD-UCD-2014-0029R03-CR_UCD_2_resource_datatype_fix</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>section 6</td>
<td>OMA-CD-UCD-2014-0031R03-CR_UCD_2_example</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>section Appendix D</td>
<td>OMA-CD-UCD-2014-0033R02-CR_UCD_2_JSON_example</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>section Appendix C</td>
<td>OMA-CD-UCD-2014-0034-CR_UCD_2_x-www_form_urlencoded_example</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>section Appendix B</td>
<td>OMA-CD-UCD-2014-0035R01-CR_UCD_2_Static_Conformance_Requirements</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>section Appendix F</td>
<td>OMA-CD-UCD-2014-0036-CR_UCD_2_Light_weight_Resources</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>section 5.3</td>
<td>OMA-CD-UCD-2014-0039R01-CR_UCD_2_Sequence_Diagrams</td>
</tr>
<tr>
<td></td>
<td>12 Jun 2014</td>
<td>section G</td>
<td>OMA-CD-UCD-2014-0042R02-CR_Interface_UCD_2_Authorization_Aspects</td>
</tr>
<tr>
<td></td>
<td>09 Sep 2014</td>
<td>All</td>
<td>OMA-CD-UCD-2014-0082R01-CR_TS_REST_ZTE</td>
</tr>
<tr>
<td></td>
<td>23 Sep 2014</td>
<td>section 6</td>
<td>OMA-CD-UCD-2014-0095-CR_TS_REST_ZTE</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-SUPPORT-S-001-M</td>
<td>Support for the RESTful UCD API</td>
<td>5, 6</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-SUPPORT-S-002-M</td>
<td>Support for the XML request &amp; response format</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-SUPPORT-S-003-M</td>
<td>Support for the JSON request &amp; response format</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-SUPPORT-S-004-O</td>
<td>Support for the application/x-www-form-urlencoded format</td>
<td>Appendix C</td>
<td></td>
</tr>
</tbody>
</table>

#### B.1.1 SCR for REST.UCD.Folder Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-FOLDER-001-M</td>
<td>Support for folder operations</td>
<td>6.1</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FOLDER-002-M</td>
<td>List folder information - GET</td>
<td>6.1.3</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FOLDER-003-M</td>
<td>Create folder - POST</td>
<td>6.1.4</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FOLDER-004-M</td>
<td>Delete folder to recycle bin or permanently - DELETE</td>
<td>6.1.6</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-FOLDER-ATTR-001-O</td>
<td>Support for the management of individual folder attributes</td>
<td>6.2</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FOLDER-ATTR-002-O</td>
<td>Retrieve individual folder attributes - GET</td>
<td>6.2.3</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FOLDER-ATTR-003-O</td>
<td>Update individual folder ACL information - PUT</td>
<td>6.2.4</td>
<td></td>
</tr>
</tbody>
</table>

#### B.1.3 SCR for REST.UCD.File Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-FILE-001-M</td>
<td>Support for file operations</td>
<td>6.3</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FILE-002-M</td>
<td>Download file - GET</td>
<td>6.3.3</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FILE-003-M</td>
<td>Upload file or Update file in range - PUT</td>
<td>6.3.4</td>
<td></td>
</tr>
<tr>
<td>Item</td>
<td>Function</td>
<td>Reference</td>
<td>Requirement</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>REST-UCD-FILE-004-M</td>
<td>Delete file to recycle bin or permanently-DELETE</td>
<td>6.3.6</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-FILE-ATTR-001-O</td>
<td>Support for the management of individual file attributes</td>
<td>6.4</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FILE-ATTR-002-O</td>
<td>Retrieve individual file attributes-GET</td>
<td>6.4.3</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-FILE-ATTR-003-O</td>
<td>Update individual file information including fileType, metadata, ACL information, sharing information - PUT</td>
<td>6.4.4</td>
<td></td>
</tr>
</tbody>
</table>

### B.1.5 SCR for REST.UCD.Recyclebin Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-RECYCLEBIN-001-M</td>
<td>Support for Recycle bin operations</td>
<td>6.5</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-RECYCLEBIN-002-M</td>
<td>List recycle bin - GET</td>
<td>6.5.3</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-RECYCLEBIN-003-M</td>
<td>Revoke or Delete the items of recycle bin-PUT</td>
<td>6.5.4</td>
<td></td>
</tr>
</tbody>
</table>

### B.1.6 SCR for REST.UCD.Search Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-SEARCH-001-M</td>
<td>Support for search operations</td>
<td>6.6</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-SEARCH-002-M</td>
<td>Search file or folder - POST</td>
<td>6.6.5</td>
<td></td>
</tr>
</tbody>
</table>

### B.1.7 SCR for REST.UCD.ListShare Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST-UCD-LISTSHARE-001-M</td>
<td>Support for listShare operations</td>
<td>6.7</td>
<td></td>
</tr>
<tr>
<td>REST-UCD-LISTSHARE-002-M</td>
<td>List all file sharing - POST</td>
<td>6.7.5</td>
<td></td>
</tr>
</tbody>
</table>
Appendix C. Application/x-www-form-urlencoded Request Format for POST Operations

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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type/Values</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromCursor</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The beginning position of the retrieve response. Omitting this value denotes the first position.\n The fromCursor is a cursor value provided by \ the server in a previous response to a request with the same search selection criteria.</td>
</tr>
<tr>
<td>maxEntries</td>
<td>xsd:int</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>searchKey</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Search key\n If there is no search key, the server will retrieval all available elements.</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>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.</td>
</tr>
</tbody>
</table>
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">

</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>
  </reference>
  <reference>
  </reference>

  <reference>
  </reference>
  <reference>
  </reference>

  <reference>
  </reference>
  </files>
</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>
</file>
  </fileAttributes>
</file>
  </fileAttributes>
  <fileType>jpg</fileType>
  <size>549778</size>
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%2B195855550100/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%2B195855550100/myfolder%2Fmydocument/mydocument1.doc"},
         {"resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B195855550100/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%2B195855550100/myfolder%2Fmydocument",
      "subFolders": {"reference": {"resourceURL": "http://example.com/exampleAPI/ucd/v1/tel%3A%2B195855550100/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"
      }
    }
  }
}}
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": [{
        "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": [{"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%2B195855550100/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 2014 08:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B195855550100/myfolder%2Fmydocument
Content-Type: application/json
Content-Length: nnnn

{"resourceReference": {"resourceURL": "http://exampleAPI/ucd/v1/tel%3A%2B195855550100/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%2B195855550100/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 2014 08:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B195855550100/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
D.10 Update individual folder ACL information (section 6.2.4.1)

Request:

```plaintext
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:

```plaintext
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%2B195855550100/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%2B195855550100/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%2B195855550100/myfolder%2Fmydocument/mydocument2.jpg
Content-Type: application/json
Content-Length: nnnn


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"
                }"
            }"
        },
    "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%2B195855550100/myfolder%2Fmydocument/mydocument1.doc HTTP/1.1
Accept: application/json
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
Host: example.com
Content-Type: binary/octet-stream
Range: bytes=start-end

[mydocument1.doc's updating content]

Response:

HTTP/1.1 200 OK
Date: Tue, 19 Jan 2014-08:30:50 GMT
Location: http://exampleAPI/ucd/v1/tel%3A%2B195855550100/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@"
    }
  },
  "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"
  }
}}
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
D.18 Update individual file’s attribute of fileType (section 6.4.4.1)

Request:

```plaintext
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:
D.19 Update individual file’s attribute of metadata (section 6.4.4.2)

Request:

```plaintext
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:

```plaintext
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"}
]}}
```
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",
}}
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

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",
      "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"
          }
        }
      ]
    }
  }
}
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.

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

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.

<table>
<thead>
<tr>
<th>Scope value</th>
<th>Description</th>
<th>For one-time access token</th>
</tr>
</thead>
<tbody>
<tr>
<td>oma_rest_ucd.all_{apiVersion}</td>
<td>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.</td>
<td>No</td>
</tr>
<tr>
<td>oma_rest_ucd.file</td>
<td>Provide access to all defined operations(e.g., read, delete, update, etc.) on file</td>
<td>No</td>
</tr>
<tr>
<td>oma_rest_ucd.folder</td>
<td>Provide access to all defined operations(e.g., read, delete, etc.) on folder</td>
<td>No</td>
</tr>
<tr>
<td>oma_rest_ucd.recycleBin</td>
<td>Provide access to all defined operations(e.g., read, delete, etc.) on recycleBin</td>
<td>No</td>
</tr>
<tr>
<td>oma_rest_ucd.search</td>
<td>Provide access to all defined operations(e.g., allowed or not allowed to search) on search</td>
<td>No</td>
</tr>
<tr>
<td>oma_rest_ucd.listShare</td>
<td>Provide access to all defined operations on listShare</td>
<td>No</td>
</tr>
</tbody>
</table>

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.
<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Section reference</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>A file</td>
<td>/{folderName}/{fileName}</td>
<td>6.3</td>
<td>GET PUT POST DELETE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>all_{apiVersion}</td>
<td>all_{apiVersion}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or file</td>
<td>or file</td>
</tr>
<tr>
<td>Individual File Attributes</td>
<td>/{folderName}/{fileName}/{ResourceRelPath}</td>
<td>6.4</td>
<td>GET PUT POST DELETE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>all_{apiVersion}</td>
<td>all_{apiVersion}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or file</td>
<td>or file</td>
</tr>
</tbody>
</table>

Table 2: Required scope values for: managing files

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Section reference</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>A folder</td>
<td>/{folderName}</td>
<td>6.1</td>
<td>GET PUT POST DELETE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>all_{apiVersion}</td>
<td>all_{apiVersion}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or folder</td>
<td>or folder</td>
</tr>
<tr>
<td>Individual folder attributes</td>
<td>/{folderName}/{ResourceRelPath}</td>
<td>6.2</td>
<td>GET PUT POST DELETE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>all_{apiVersion}</td>
<td>all_{apiVersion}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or folder</td>
<td>or folder</td>
</tr>
</tbody>
</table>

Table 3: Required scope values for: managing folder

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Section reference</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recycle Bin</td>
<td>/recycleBin</td>
<td>6.5</td>
<td>GET PUT POST DELETE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>all_{apiVersion}</td>
<td>all_{apiVersion}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or recycleBin</td>
<td>or recycleBin</td>
</tr>
</tbody>
</table>

Table 4: Required scope values for: managing recyclebin
<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Section reference</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>GET</td>
</tr>
<tr>
<td>File or Folder search</td>
<td>/operations/search</td>
<td>6.6</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Table 5: Required scope values for: managing search

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/ucd/{apiVersion}/{userId}</th>
<th>Section reference</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>GET</td>
</tr>
<tr>
<td>List shared files</td>
<td>/operations/listShare</td>
<td>6.7</td>
<td>n/a</td>
</tr>
</tbody>
</table>

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’.