

1



2

Partial Upload CDMI Extension

Version 2.0

3

4

5 ABSTRACT: This CDMI Extension is intended for developers who are considering a standardized way to add
6 functionality to CDMI. When multiple compatible implementations are demonstrated and approved by the Technical
7 Working Group, this extension will be incorporated into the CDMI standard.

8 This document has been released and approved by the SNIA. The SNIA believes that the ideas, methodologies, and
9 technologies described in this document accurately represent the SNIA goals and are appropriate for widespread
10 distribution. Suggestion for revision should be directed to <http://www.snia.org/feedback/>.

11

SNIA Working Draft

12

January 29, 2021

USAGE

Copyright © 2021 SNIA. All rights reserved. All other trademarks or registered trademarks are the property of their respective owners.

The SNIA hereby grants permission for individuals to use this document for personal use only, and for corporations and other business entities to use this document for internal use only (including internal copying, distribution, and display) provided that:

1. Any text, diagram, chart, table or definition reproduced shall be reproduced in its entirety with no alteration, and,

2. Any document, printed or electronic, in which material from this document (or any portion hereof) is reproduced shall acknowledge the SNIA copyright on that material, and shall credit the SNIA for granting permission for its reuse.

Other than as explicitly provided above, you may not make any commercial use of this document, sell any excerpt or this entire document, or distribute this document to third parties. All rights not explicitly granted are expressly reserved to SNIA.

Permission to use this document for purposes other than those enumerated above may be requested by emailing tcmd@snia.org. Please include the identity of the requesting individual or company and a brief description of the purpose, nature, and scope of the requested use.

All code fragments, scripts, data tables, and sample code in this SNIA document are made available under the following license:

BSD 3-Clause Software License

Copyright (c) 2021, The Storage Networking Industry Association.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of The Storage Networking Industry Association (SNIA) nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

DISCLAIMER

The information contained in this publication is subject to change without notice. The SNIA makes no warranty of any kind with regard to this specification, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The SNIA shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this specification.

Suggestions for revisions should be directed to <https://www.snia.org/feedback/>.

Copyright © 2021 SNIA. All rights reserved. All other trademarks or registered trademarks are the property of their respective owners.

Contents

56

57	Clause 1: Partial Upload CDMI Extension	1
58	1.1 Overview	1
59	1.2 Instructions to the Editor	1
60	1.3 Overview	1
61	1.4 Examples	2

Clause 1

Partial Upload CDMI Extension

1.1 Overview

CDMI 2.0.0 provides limited support for partial uploads. The following extension to the X-CDMI-Partial header is proposed to support managing conflicting uploads and specifying completion conditions for parallel uploads:

```
X-CDMI-Partial: [ true | false ] | [ upload-id=<upload-id> [ ; [ count=<integer> |  
range=<byte-range> ] ] [ ; replace= { true | false } ] ]
```

Partial uploads by multiple concurrent clients are accommodated by specifying a unique “upload-id” for each set of partial uploads.

A condition is associated with an upload id to indicate when the partial upload will be considered complete. The condition may be associated with an upload id at any time. Once a condition is associated with an upload id, specifying a different condition is considered an error.

A count condition handles the scenario when a partial upload is to be completed when a specific number of partial uploads are received by the server.

A range condition handles the scenario when a partial upload is to be completed when the server receives a specific byte range.

Replace indicates if the partial uploads replace the entire object or just the specified byte ranges.

When partial uploads associated with an upload id have not completed within a given time, the upload id will time out, and the partial uploads associated with that upload id will be discarded by the server.

1.2 Instructions to the Editor

To merge this extension into the CDMI 2.0.0 specification, make the following changes:

1. Add a new section 5.9 titled “Partial Uploads”

1.3 Overview

CDMI defines a custom “X-CDMI-Partial” header that indicates when a partial upload is being performed. The value of this header is formatted according to the following BNF:

```
X-CDMI-Partial: false | true | upload-id=<upload-id> [ [ ;count=<integer> ] | [ ;  
range=<byte-range> ] ] [ ;replace=( true | false ) ]
```

- “false” (or header not present) – Indicates that the set of uploads associated with a null upload ID shall be considered complete, and the object shall be updated.
- “true” – Indicates that the newly created object is part of a series of uploads and the value has not yet been fully populated. These uploads are considered to have a null upload ID.
- “upload-id” – Indicates that requests with the same upload ID are part of the set of partial uploads.

- “count” – An integer value greater than zero that indicates that when exactly this many partial uploads with the same upload ID are received, the set of uploads is considered complete, and the object shall be updated. If the number of received uploads is larger than the specified count, an HTTP status code of 400 Bad Request shall be returned.
- “range” – A byte range as specified in section 14.35.1 of [rfc2616] that indicates that when a given byte range of partial uploads with the same upload ID are received, the set of uploads is considered complete, and the object shall be updated.
- “replace” – When multiple ranges are sent as part of a partial upload, if this flag has the value “true”, this indicates that the entire object shall be replaced by the set of ranged uploads (with any range gaps zero-filled). If it has the value “false”, the set of uploads shall be used to update the object, not replace it.

For a given upload ID, if a condition or replace flag is received that is different from a previously received condition or replace flag, an HTTP status code of 400 Bad Request shall be returned.

For a given upload ID, if a partial upload is received with an `content-range` exactly the same as a previously received `content-range`, the value associated with that range shall be replaced with the newer value. This allows partial uploads to be retried without error. In this case, the count shall not change.

For a given upload ID, if a partial upload is received with a `content-range` that overlaps a previously received `content-range` and is not exactly the same as a previously received `contentrange`, an HTTP status code of 400 Bad Request shall be returned.

If a set of uploads for a particular upload ID is not complete and no messages for that upload ID are received before the timeout value specified in the `cdmi_partial_timeout` expires, then the server shall terminate the set and discard all previously received partial ranges for that upload ID.

If a new object is being created using the `X-CDMI-Partial` header, the `completionStatus` field in the response body shall be set to “Processing” and the value of the object shall not be returned to clients until the partial upload is considered complete.

If an existing object is being updated or replaced using the `X-CDMI-Partial` header, the object shall not be updated until the partial upload is considered complete.

1.4 Examples

EXAMPLE 1: No-op Partial Upload (same as if `X-CDMI-Partial` not included):

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 37
--> X-CDMI-Partial: false
-->
--> This is the Value of this Data Object

<-- HTTP/1.1 201 Created
```

EXAMPLE 2: Basic Partial Upload (Single client sending subsequent PUTs after receiving indication of prior successful PUT to append data):

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 37
--> X-CDMI-Partial: true
-->
--> This is the Value of this Data Object

<-- HTTP/1.1 202 Accepted
```

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 13
--> X-CDMI-Partial: false
```

(continues on next page)

(continued from previous page)

```
-->
--> in two parts.

<-- HTTP/1.1 201 Created
```

124 **EXAMPLE 3: Range-based Partial Upload (Single client sending concurrent / un-ordered PUTs, with final PUT when all**
125 **ranges succesfully stored):**

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 37
--> Content-Range: 0-36
--> X-CDMI-Partial: true
-->
--> This is the Value of this Data Object

<-- HTTP/1.1 202 Accepted
```

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 13
--> Content-Range: 37-49
--> X-CDMI-Partial: true
-->
--> in two parts.

<-- HTTP/1.1 202 Accepted
```

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 0
--> X-CDMI-Partial: false
-->
-->

<-- HTTP/1.1 201 Created
```

126 **EXAMPLE 4: Range-based Partial Upload (Multiple client sending concurrent / un-ordered PUTs, with final PUT when**
127 **all ranges succesfully stored):**

128 **Client 1:**

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 37
--> Content-Range: 0-36
--> X-CDMI-Partial: upload-id=8723648734
-->
--> This is the Value of this Data Object

<-- HTTP/1.1 202 Accepted
```

129 **Client 2:**

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 13
--> Content-Range: 37-49
--> X-CDMI-Partial: upload-id=8723648734
-->
--> in two parts.

<-- HTTP/1.1 202 Accepted
```

130 Client 3:

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 0
--> X-CDMI-Partial: upload-id=8723648734
-->
-->
<-- HTTP/1.1 201 Created
```

131 **EXAMPLE 5: Range-based Partial Upload (Multiple client sending concurrent / un-ordered PUTs, with object finalized**
132 **when a count is reached):**

133 Client 1:

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 37
--> Content-Range: 0-36
--> X-CDMI-Partial: upload-id=8723648734; count=2
-->
--> This is the Value of this Data Object
<-- HTTP/1.1 202 Accepted
```

134 Client 2:

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 13
--> Content-Range: 37-49
--> X-CDMI-Partial: upload-id=8723648734; count=2
-->
--> in two parts.
<-- HTTP/1.1 201 Created
```

135 **EXAMPLE 6: Range-based Partial Upload (Multiple client sending concurrent / un-ordered PUTs, with object finalized**
136 **when a range is received):**

137 Client 1:

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 37
--> Content-Range: 0-36
--> X-CDMI-Partial: upload-id=8723648734; range=0-49
-->
--> This is the Value of this Data Object
<-- HTTP/1.1 202 Accepted
```

138 Client 2:

```
--> PUT /cdmi/2.0.0/MyContainer/MyDataObject.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 13
--> Content-Range: 37-49
--> X-CDMI-Partial: upload-id=8723648734; ; range=0-49
-->
--> in two parts.
<-- HTTP/1.1 201 Created
```


2. Add the following entries to the end of the table starting on line 135 of `cdmi_advanced/cdmi_capability_object.txt`, as follows:

Table 1: System-wide capabilities

Capability name	Type	Definition
<code>cdmi_partial</code>	JSON string	If present and “true”, indicates that the cloud storage system shall support the X-CDMI-Partial header “true” and “false” values.
<code>cdmi_partial_uploadid</code>	JSON string	If present and “true”, indicates that the cloud storage system shall support the X-CDMI-Partial header upload-id values.
<code>cdmi_partial_count</code>	JSON string	If present and “true”, indicates that the cloud storage system shall support the X-CDMI-Partial header count completion condition.
<code>cdmi_partial_range</code>	JSON string	If present and “true”, indicates that the cloud storage system shall support the X-CDMI-Partial header range completion condition.
<code>cdmi_partial_replace</code>	JSON string	If present and “true”, indicates that the cloud storage system shall support the X-CDMI-Partial header replace flag.
<code>cdmi_partial_timeout</code>	JSON string	If present, this capability indicates the upload-id timeout duration in seconds.

8. Replace the description for the “X-CDMI-Partial” table row in tables 9, 16, 26, 30, 41 and 67 as follows. Indicates that a partial upload is being performed. See [Section 1.3](#).