BS ISO/IEC 17826:2012

BSI Standards Publication

Information technology — Cloud Data Management Interface (CDMI)

bsi.

...making excellence a habit.™

Document Sharing Hub

National foreword

This British Standard is the UK implementation of ISO/IEC 17826:2012.

The UK participation in its preparation was entrusted to Technical Committee ICT/-/1, Information systems co-ordination.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2013.
Published by BSI Standards Limited 2013

ISBN 978 0 580 76991 7
ICS 35.040

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 May 2013.

Amendments issued since publication

Date	Text affected

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

INTERNATIONAL STANDARD

ISO/IEC 17826

First edition
2012-11-15

Information technology — Cloud Data Management Interface (CDMI)

Technologies de l'information — Interface de management des données du nuage informatique (CDMI)

Reference number
ISO/IEC 17826:2012(E)

Documents
Document Sharing Hub

© ISO/IEC 2012

Full document details: BS ISO/IEC 17826:2012

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2012

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

© ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

International Standard

ISO/IEC

Information Technology - Cloud Data Management Interface (CDMI™)

Reference Number
ISO/IEC 17826:2012(E)

November 15, 2012

Documents
Document Sharing Hub

PATENT STATEMENT

CAUTION: The developers of this international standard have requested that holders of patents that may be required for the implementation of the standard, disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which, if any, patents may apply to this international standard. As of the date of publication of this international standard, following calls for the identification of patents that may be required for the implementation of the standard, no such claims have been made. No further patent search is conducted by the developer or the publisher in respect to any standard it processes. No representation is made or implied that licenses are not required to avoid infringement in the use of this international standard.

Copyright Notice

This ISO document is copyright-protected by ISO. While the reproduction of working drafts or committee drafts in any form for use by participants in the ISO standards development process is permitted without prior permission from ISO, neither this document nor any extract from it may be reproduced, stored or transmitted in any form for any other purpose without prior written permission from ISO.

Requests for permission to reproduce this document for the purpose of selling it should be addressed as shown below or to ISO's member body in the country of the requester:

The Storage Networking Industry Association
425 Market Street, Suite 1020
San Francisco, CA 94105

Reproduction for sales purposes may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

CDMI is a trademark of the Storage Networking Industry Association.

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

Contents

Foreword 15
Introduction 16

1 Scope 18

2 Normative References 18

3 Terms 20

4 Conventions 23

4.1 Interface Format 23
4.2 Typographical Conventions 23
4.3 Request and Response Body Requirements 23
4.4 Key Word Requirements 24

5 Overview of Cloud Storage 25

5.1 Introduction 25
5.2 What is Cloud Storage? 25
5.3 Data Storage as a Service 25
5.4 Data Management for Cloud Storage 27
5.5 Data and Container Management 27
5.6 Reference Model for Cloud Storage Interfaces 28
5.7 Cloud Data Management Interface 29
5.8 Object Model for CDMI 30
5.9 CDMI Metadata 31
5.10 Object ID 32
5.11 CDMI Object ID Format 32
5.12 Security 33
5.13 Required HTTP Support 34
5.13.1 RFC 2616 Support Requirements 34
5.13.2 Content-Type Negotiation 34
5.13.3 Range Support 34
5.13.4 URI Escaping 34
5.13.5 Use of URIs 35
5.13.6 Reserved Characters 35
5.14 Time Representations 35
5.15 Backwards Compatibility 36
5.15.1 Value Transfer Encoding 36
5.15.2 Container Export Capabilities 36

6 Common Operations 37

6.1 Overview 37
6.2 Discover the Capabilities of a Cloud Storage Provider 37
6.3 Create a New Container 38
6.4 Create a Data Object in a Container 38
6.5 List the Contents of a Container 39
6.6 Read the Contents of a Data Object 39
6.7 Read Only the Value of a Data Object 40
6.8 Delete a Data Object 40

7 Interface Standard 41

7.1 HTTP Status Codes 41
7.2 Object References 41

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

8 Data Object Resource Operations 43
8.1 Overview 43
8.1.1 Data Object Metadata 44
8.1.2 Data Object Consistency 44
8.1.3 Data Object Representations 44
8.2 Create a Data Object Using CDMI Content Type 44
8.2.1 Synopsis 44
8.2.2 Delayed Completion of Create 45
8.2.3 Capabilities 45
8.2.4 Request Headers 46
8.2.5 Request Message Body 46
8.2.6 Response Headers 48
8.2.7 Response Message Body 49
8.2.8 Response Status 50
8.2.9 Examples 50
8.3 Create a Data Object using a Non-CDMI Content Type 51
8.3.1 Synopsis 51
8.3.2 Capability 51
8.3.3 Request Headers 52
8.3.4 Request Message Body 52
8.3.5 Response Headers 52
8.3.6 Response Message Body 52
8.3.7 Response Status 52
8.3.8 Example 53
8.4 Read a Data Object using CDMI Content Type 53
8.4.1 Synopsis 53
8.4.2 Capabilities 53
8.4.3 Request Headers 54
8.4.4 Request Message Body 54
8.4.5 Response Headers 54
8.4.6 Response Message Body 54
8.4.7 Response Status 57
8.4.8 Examples 57
8.5 Read a Data Object using a Non-CDMI Content Type 59
8.5.1 Synopsis 59
8.5.2 Capabilities 59
8.5.3 Request Header 59
8.5.4 Request Message Body 59
8.5.5 Response Headers 59
8.5.6 Response Message Body 60
8.5.7 Response Status 60
8.5.8 Examples 60
8.6 Update a Data Object using CDMI Content Type 61
8.6.1 Synopsis 61
8.6.2 Capabilities 61
8.6.3 Request Headers 62
8.6.4 Request Message Body 62
8.6.5 Response Header 64
8.6.6 Response Message Body 64
8.6.7 Response Status 64
8.6.8 Examples 65
8.7 Update a Data Object using a Non-CDMI Content Type 66
8.7.1 Synopsis 66
8.7.2 Capabilities 67
8.7.3 Request Headers 67
8.7.4 Request Message Body 67
8.7.5 Response Header 67
8.7.6 Response Message Body 68

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

8.7.7 Response Status ... 68
8.7.8 Examples ... 68
8.8 Delete a Data Object using CDMI Content Type ... 69
8.8.1 Synopsis ... 69
8.8.2 Capability ... 69
8.8.3 Request Header ... 69
8.8.4 Request Message Body ... 69
8.8.5 Response Headers ... 69
8.8.6 Response Message Body ... 69
8.8.7 Response Status ... 70
8.8.8 Example ... 70
8.9 Delete a Data Object using a Non-CDMI Content Type ... 70
8.9.1 Synopsis ... 70
8.9.2 Capability ... 70
8.9.3 Request Headers ... 70
8.9.4 Request Message Body ... 71
8.9.5 Response Headers ... 71
8.9.6 Response Message Body ... 71
8.9.7 Response Status ... 71
8.9.8 Example ... 71

9 Container Object Resource Operations ... 72

9.1 Overview ... 72
9.1.1 Container Metadata ... 73
9.1.2 Reserved Container Names ... 73
9.1.3 Container Object Addressing ... 73
9.1.4 Container Object Representations ... 73
9.2 Create a Container Object using CDMI Content Type ... 74
9.2.1 Synopsis ... 74
9.2.2 Delayed Completion of Create ... 74
9.2.3 Capabilities ... 74
9.2.4 Request Headers ... 75
9.2.5 Request Message Body ... 75
9.2.6 Response Headers ... 77
9.2.7 Response Message Body ... 77
9.2.8 Response Status ... 78
9.2.9 Example ... 79
9.3 Create a Container Object using a Non-CDMI Content Type ... 80
9.3.1 Synopsis ... 80
9.3.2 Capability ... 80
9.3.3 Request Headers ... 80
9.3.4 Request Message Body ... 80
9.3.5 Response Headers ... 80
9.3.6 Response Message Body ... 80
9.3.7 Response Status ... 81
9.3.8 Example ... 81
9.4 Read a Container Object using CDMI Content Type ... 81
9.4.1 Synopsis ... 81
9.4.2 Capabilities ... 81
9.4.3 Request Headers ... 82
9.4.4 Request Message Body ... 82
9.4.5 Response Headers ... 82
9.4.6 Response Message Body ... 83
9.4.7 Response Status ... 84
9.4.8 Examples ... 85
9.5 Update a Container Object using CDMI Content Type ... 86
9.5.1 Synopsis ... 86
9.5.2 Delayed Completion of Snapshot ... 87

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

9.5.3 Capabilities ... 87
9.5.4 Request Headers ... 87
9.5.5 Request Message Body ... 88
9.5.6 Response Header ... 89
9.5.7 Response Message Body ... 89
9.5.8 Response Status ... 89
9.5.9 Examples ... 90
9.6 Delete a Container Object using CDMI Content Type ... 91
9.6.1 Synopsis ... 91
9.6.2 Capability ... 91
9.6.3 Request Header ... 91
9.6.4 Request Message Body ... 91
9.6.5 Response Headers ... 91
9.6.6 Response Message Body ... 91
9.6.7 Response Status ... 92
9.6.8 Example ... 92
9.7 Delete a Container Object using a Non-CDMI Content Type ... 92
9.7.1 Synopsis ... 92
9.7.2 Capability ... 92
9.7.3 Request Headers ... 92
9.7.4 Request Message Body ... 93
9.7.5 Response Headers ... 93
9.7.6 Response Message Body ... 93
9.7.7 Response Status ... 93
9.7.8 Example ... 93
9.8 Create (POST) a New Data Object using CDMI Content Type ... 93
9.8.1 Synopsis ... 93
9.8.2 Delayed Completion of Create ... 94
9.8.3 Capabilities ... 94
9.8.4 Request Headers ... 95
9.8.5 Request Message Body ... 96
9.8.6 Response Headers ... 98
9.8.7 Response Message Body ... 98
9.8.8 Response Status ... 99
9.8.9 Examples ... 100
9.9 Create (POST) a New Data Object using a Non-CDMI Content Type ... 101
9.9.1 Synopsis ... 101
9.9.2 Capability ... 101
9.9.3 Request Header ... 101
9.9.4 Request Message Body ... 101
9.9.5 Response Header ... 102
9.9.6 Response Message Body ... 102
9.9.7 Response Status ... 102
9.9.8 Examples ... 102
9.10 Create (POST) a New Queue Object using CDMI Content Type ... 103
9.10.1 Synopsis ... 103
9.10.2 Delayed Completion of Create ... 103
9.10.3 Capabilities ... 104
9.10.4 Request Headers ... 104
9.10.5 Request Message Body ... 105
9.10.6 Response Headers ... 106
9.10.7 Response Message Body ... 106
9.10.8 Response Status ... 107
9.10.9 Example ... 108

10 Domain Object Resource Operations ... 109

10.1 Overview ... 109
10.1.1 Domain Object Metadata ... 109

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

10.1.2 Domain Object Summaries 109
10.1.3 Domain Object Membership 112
10.1.4 Domain Usage in Access Control 114
10.1.5 Domain Object Representations 115
10.2 Create a Domain Object using CDMI Content Type 115
10.2.1 Synopsis 115
10.2.2 Capabilities 115
10.2.3 Request Headers 115
10.2.4 Request Message Body 116
10.2.5 Response Headers 117
10.2.6 Response Message Body 117
10.2.7 Response Status 118
10.2.8 Example 118
10.3 Read a Domain Object using CDMI Content Type 119
10.3.1 Synopsis 119
10.3.2 Capabilities 119
10.3.3 Request Headers 119
10.3.4 Request Message Body 119
10.3.5 Response Headers 120
10.3.6 Response Message Body 120
10.3.7 Response Status 121
10.3.8 Examples 121
10.4 Update a Domain Object using CDMI Content Type 122
10.4.1 Synopsis 122
10.4.2 Capability 123
10.4.3 Request Headers 123
10.4.4 Request Message Body 123
10.4.5 Response Header 124
10.4.6 Response Message Body 124
10.4.7 Response Status 124
10.4.8 Example 125
10.5 Delete a Domain Object using CDMI Content Type 125
10.5.1 Synopsis 125
10.5.2 Capability 125
10.5.3 Request Headers 125
10.5.4 Request Message Body 125
10.5.5 Response Headers 126
10.5.6 Response Message Body 126
10.5.7 Response Status 126
10.5.8 Example 126

11 Queue Object Resource Operations 127

11.1 Overview 127
11.1.1 Queue Object Metadata 128
11.1.2 Queue Object Addressing 128
11.1.3 Queue Object Representations 128
11.2 Create a Queue Object using CDMI Content Type 128
11.2.1 Synopsis 128
11.2.2 Delayed Completion of Create: 128
11.2.3 Capabilities 129
11.2.4 Request Headers 129
11.2.5 Request Message Body 130
11.2.6 Response Headers 131
11.2.7 Response Message Body 131
11.2.8 Response Status 132
11.2.9 Example 132
11.3 Read a Queue Object using CDMI Content Type 133
11.3.1 Synopsis 133

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

11.3.2 Capabilities 134
11.3.3 Request Headers 134
11.3.4 Request Message Body 134
11.3.5 Response Headers 134
11.3.6 Response Message Body 135
11.3.7 Response Status 137
11.3.8 Examples 137
11.4 Update a Queue Object using CDMI Content Type 139
11.4.1 Synopsis 139
11.4.2 Capability 139
11.4.3 Request Headers 140
11.4.4 Request Message Body 140
11.4.5 Response Header 141
11.4.6 Response Message Body 141
11.4.7 Response Status 141
11.4.8 Example 141
11.5 Delete a Queue Object using CDMI Content Type 142
11.5.1 Synopsis 142
11.5.2 Capability 142
11.5.3 Request Header 142
11.5.4 Request Message Body 142
11.5.5 Response Headers 142
11.5.6 Response Message Body 142
11.5.7 Response Status 143
11.5.8 Example 143
11.6 Enqueue a New Queue Value using CDMI Content Type 143
11.6.1 Synopsis 143
11.6.2 Capability 143
11.6.3 Request Headers 144
11.6.4 Request Message Body 144
11.6.5 Response Headers 145
11.6.6 Response Message Body 145
11.6.7 Response Status 146
11.6.8 Examples 146
11.7 Delete a Queue Object Value using CDMI Content Type 148
11.7.1 Synopsis 148
11.7.2 Capability 148
11.7.3 Request Header 148
11.7.4 Request Message Body 148
11.7.5 Response Headers 148
11.7.6 Response Message Body 148
11.7.7 Response Status 149
11.7.8 Example 149

12 Capability Object Resource Operations 150

12.1 Overview 150
12.1.1 Cloud Storage System-Wide Capabilities 151
12.1.2 Storage System Metadata Capabilities 154
12.1.3 Data System Metadata Capabilities 154
12.1.4 Data Object Capabilities 157
12.1.5 Container Capabilities 157
12.1.6 Domain Object Capabilities 159
12.1.7 Queue Object Capabilities 160
12.1.8 Capability Object Representations 160
12.2 Read a Capabilities Object using CDMI Content Type 160
12.2.1 Synopsis 160
12.2.2 Capability 161
12.2.3 Request Headers 161

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

12.2.4 Request Message Body 161
12.2.5 Response Headers 161
12.2.6 Response Message Body 162
12.2.7 Response Status 162
12.2.8 Examples 163

13 Exported Protocols 165

13.1 Overview 165
13.2 Exported Protocol Structure 166
13.2.1 Mapping Names from CDMI to Another Protocol 167
13.2.1.1 Capabilities 167
13.2.1.2 Domains 167
13.2.1.3 Caching 167
13.2.1.4 Groups 168
13.2.1.5 Synopsis 168
13.2.2 Administrative Users 169
13.2.3 User and Groupname Mapping Syntax and Evaluation Rules 169
13.3 Discovering and Mounting Containers via Foreign Protocols 170
13.4 NFS Exported Protocol 171
13.5 CIFS Exported Protocol 173
13.6 OCCI Exported Protocol 173
13.7 iSCSI Export Modifications 174
13.7.1 Read Container 174
13.7.2 Create and Update Containers 174
13.7.3 Modify an Export 175
13.8 WebDAV Exported Protocol 175

14 Snapshots 176

15 Serialization/Deserialization 177

15.1 Overview 177
15.2 Exporting Serialized Data 177
15.3 Importing Serialized Data 177
15.3.1 Canonical Format 178
15.3.2 Example JSON Canonical Serialized Format 178

16 Metadata 180

16.1 Access Control 180
16.1.1 ACL and ACE Structure 180
16.1.2 ACE Types 180
16.1.3 ACE Who 180
16.1.4 ACE Flags 181
16.1.5 ACE Mask Bits 182
16.1.6 ACL Evaluation 183
16.1.7 Example ACE Mask Expressions 185
16.1.8 Canonical Format for ACE Hexadecimal Quantities 185
16.1.9 JSON Format for ACLs 186
16.2 Support for User Metadata 187
16.3 Support for Storage System Metadata 187
16.4 Support for Data System Metadata 188
16.5 Support for Provided Data System Metadata 193

17 Retention and Hold Management 195

17.1 Introduction 195
17.2 Retention Management Disciplines 195
17.3 CDMI Retention 195
17.4 CDMI Hold 196
17.5 CDMI Auto-deletion 198

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

17.6 Retention Security Considerations 198

18 Scope Specification 199

18.1 Introduction 199
18.2 Examples 199
18.3 Query Matching Expressions 200

19 Results Specification 205

19.1 Introduction 205
19.2 Examples 205

20 Logging 207

20.1 Overview 207
20.2 Object Logging 207
20.3 Security Logging 207
20.4 Data Management Logging 208
20.5 Logging Queues 208
20.6 Logging Security Considerations 210

21 Notification Queues 211

22 Query Queues 215

22.1 Overview 215
22.2 Extending CDMI Query 217

Annex A
(normative)
Transport Security 218

A.1 Introduction 218
A.2 General Requirements for HTTP Implementations 218
A.3 Basic HTTP Security 219
A.4 HTTP over TLS (HTTPS) 219
A.5 Transport Layer Security (TLS) 219
A.5.1 Cipher Suites 220
A.5.2 Digital Certificates 220

Annex B
(informative)
Bibliography 224

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

Figures

Figure 1 – Existing Data Storage Interface Standards 26
Figure 2 – Storage Interfaces for Object Storage Client Data 27
Figure 3 – Cloud Storage Reference Model 28
Figure 4 – CDMI Object Model 30
Figure 5 – Object Transitions between Named and ID-only 31
Figure 6 – Object ID Format 32
Figure 7 – Hierarchy of Capabilities 150
Figure 8 – CDMI and OCCI in an Integrated Cloud Computing Environment 165
Figure 9 – Snapshot Container Structure 176
Figure 10 – Object Retention 196
Figure 11 – Object Hold 197
Figure 12 – Object Hold on Object with Retention 197
Figure 13 – Object with Multiple Holds 197

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

Tables

Table 1 – Interface Format ... 23
Table 2 – Key Word Requirements ... 24
Table 3 – Types of Resources in the Model ... 30
Table 4 – Creation/Consumption of Storage System Metadata ... 31
Table 5 – Relative URIs Resolved Against Root URIs ... 35
Table 6 – HTTP Status Codes ... 41
Table 7 – Request Headers for Creating a CDMI Data Object using CDMI Content Type ... 46
Table 8 – Request Message Body - Create a Data Object using CDMI Content Type ... 46
Table 9 – Response Headers - Create a Data Object using CDMI Content Type ... 48
Table 10 – Response Message Body - Create a Data Object using CDMI Content Type ... 49
Table 11 – HTTP Status Codes - Create a Data Object using CDMI Content Type ... 50
Table 12 – Request Headers - Create a CDMI Data Object using a Non-CDMI Content Type ... 52
Table 13 – HTTP Status Codes - Create a Data Object using a Non-CDMI Content Type ... 52
Table 14 – Request Headers - Read a CDMI Data Object using CDMI Content Type ... 54
Table 15 – Response Headers - Read a CDMI Data Object using CDMI Content Type ... 54
Table 16 – Response Message Body - Read a Data Object using CDMI Content Type ... 54
Table 17 – HTTP Status Codes - Read a CDMI Data Object using CDMI Content Type ... 57
Table 18 – Request Header - Read a CDMI Data Object using a Non-CDMI Content Type ... 59
Table 19 – Response Headers - Read a CDMI Data Object using a Non-CDMI Content Type ... 59
Table 20 – HTTP Status Codes - Read a CDMI Data Object using a Non-CDMI Content Type ... 60
Table 21 – Request Headers - Update a CDMI Data Object using CDMI Content Type ... 62
Table 22 – Request Message Body - Update a CDMI Data Object using CDMI Content Type ... 62
Table 23 – Response Header - Update a CDMI Data Object using CDMI Content Type ... 64
Table 24 – HTTP Status Codes - Update a CDMI Data Object using CDMI Content Type ... 64
Table 25 – Request Headers - Update a CDMI Data Object using a Non-CDMI Content Type ... 67
Table 26 – Response Header - Update a CDMI Data Object using a Non-CDMI Content Type ... 67
Table 27 – HTTP Status Codes - Update a CDMI Data Object using a Non-CDMI Content Type ... 68
Table 28 – Request Header - Delete a CDMI Data Object using CDMI Content Type ... 69
Table 29 – HTTP Status Codes - Delete a CDMI Data Object using CDMI Content Type ... 70
Table 30 – HTTP Status Codes - Delete a CDMI Data Object using a Non-CDMI Content Type ... 71
Table 31 – Container Metadata ... 73
Table 32 – Request Headers - Create a Container Object using CDMI Content Type ... 75
Table 33 – Request Message Body - Create a Container Object using CDMI Content Type ... 75
Table 34 – Response Headers - Create a Container Object using CDMI Content Type ... 77
Table 35 – Response Message Body - Create a Container Object using CDMI Content Type ... 77
Table 36 – HTTP Status Codes - Create a CDMI Container Object using CDMI Content Type ... 78
Table 37 – HTTP Status Codes - Create a Container Object using a Non-CDMI Content Type ... 81
Table 38 – Request Headers - Read a Container Object using CDMI Content Type ... 82
Table 39 – Response Headers - Read a Container Object using CDMI Content Type ... 82
Table 40 – Response Message Body - Read a Container Object using CDMI Content Type ... 83
Table 41 – HTTP Status Codes - Read a Container Object using CDMI Content Type ... 84
Table 42 – Request Headers - Update a Container Object using CDMI Content Type ... 87
Table 43 – Request Message Body - Update a Container Object using CDMI Content Type ... 88
Table 44 – Response Header - Update a Container Object using CDMI Content Type ... 89
Table 45 – HTTP Status Codes - Update a Container Object using CDMI Content Type ... 89
Table 46 – Request Header - Delete a Container Object using CDMI Content Type ... 91
Table 47 – HTTP Status Codes - Delete a Container Object using CDMI Content Type ... 92
Table 48 – HTTP Status Codes - Delete a Container Object using a Non-CDMI Content Type ... 93
Table 49 – Request Headers - Create a New Data Object using CDMI Content Type ... 95
Table 50 – Request Message Body - Create a New Data Object using CDMI Content Type ... 96
Table 51 – Response Headers - Create a New Data Object using CDMI Content Type ... 98
Table 52 – Response Message Body - Create a New Data Object using CDMI Content Type ... 98
Table 53 – HTTP Status Codes - Create a New Data Object using CDMI Content Type ... 99
Table 54 – Request Header - Create a New Data Object using a Non-CDMI Content Type ... 101
Table 55 – Response Header - Create a New Data Object using a Non-CDMI Content Type ... 102
Table 56 – HTTP Status Codes - Create a New Data Object using a Non-CDMI Content Type ... 102

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

Table 57 – Request Headers - Create a New Queue Object using CDMI Content Type 104
Table 58 – Request Message Body - Create a New Queue Object using CDMI Content Type..... 105
Table 59 – Response Headers - Create a New CDMI Queue Object using CDMI Content Type..... 106
Table 60 – Response Message Body - Create a New Queue Object with CDMI Content 106
Table 61 – HTTP Status Codes - Create a New CDMI Queue Object using CDMI Content Type..... 107
Table 62 – Required Metadata for a Domain Object 109
Table 63 – Contents of Domain Summary Objects 111
Table 64 – Required Settings for Domain Member User Objects 113
Table 65 – Required Settings for Domain Member Delegation Objects..... 114
Table 66 – Request Headers - Create a Domain Object using CDMI Content Type 115
Table 67 – Request Message Body - Create a Domain Object using CDMI Content Type 116
Table 68 – Response Headers - Create a Domain Object using CDMI Content Type..... 117
Table 69 – Response Message Body - Create a Domain Object using CDMI Content Type 117
Table 70 – HTTP Status Codes - Create a Domain Object using CDMI Content Type 118
Table 71 – Request Headers - Read a Domain Object using CDMI Content Type..... 119
Table 72 – Response Headers - Read a Domain Object using CDMI Content Type..... 120
Table 73 – Response Message Body - Read a Domain Object using CDMI Content Type..... 120
Table 74 – HTTP Status Codes - Read a Domain Object using CDMI Content Type..... 121
Table 75 – Request Headers - Update a Domain Object using CDMI Content Type..... 123
Table 76 – Request Message Body - Update a Domain Object using CDMI Content Type 123
Table 77 – Response Header - Update a Domain Object using CDMI Content Type..... 124
Table 78 – HTTP Status Codes - Update a Domain Object using CDMI Content Type..... 124
Table 79 – Request Headers - Delete a Domain Object using CDMI Content Type..... 125
Table 80 – HTTP Status Codes - Delete a Domain Object using CDMI Content Type..... 126
Table 81 – Request Headers - Create a Queue Object using CDMI Content Type 129
Table 82 – Request Message Body - Create a Queue Object using CDMI Content Type..... 130
Table 83 – Response Headers - Create a Queue Object using CDMI Content Type 131
Table 84 – Response Message Body - Create a Queue Object using CDMI Content Type 131
Table 85 – HTTP Status Codes - Create a Queue Object using CDMI Content Type 132
Table 86 – Request Headers - Read a Queue Object using CDMI Content Type 134
Table 87 – Response Headers - Read a Queue Object using CDMI Content Type 134
Table 88 – Response Message Body - Read a Queue Object using CDMI Content Type 135
Table 89 – HTTP Status Codes - Read a Queue Object using CDMI Content Type 137
Table 90 – Request Headers - Update a Queue Object using CDMI Content Type 140
Table 91 – Request Message Body - Update a Queue Object using CDMI Content Type 140
Table 92 – Response Header - Update a Queue Object using CDMI Content Type 141
Table 93 – HTTP Status Codes - Update a Queue Object using CDMI Content Type 141
Table 94 – Request Header - Delete a Queue Object using CDMI Content Type 142
Table 95 – HTTP Status Codes - Delete a Queue Object using CDMI Content Type 143
Table 96 – Request Headers - Enqueue a New Queue Object Value using CDMI Content Type 144
Table 97 – Request Message Body - Enqueue a New Queue Value using CDMI Content Type 144
Table 98 – HTTP Status Codes - Enqueue a New Queue Object Value using CDMI Content Type..... 146
Table 99 – Request Header - Delete a Queue Object Value using CDMI Content Type 148
Table 100 – HTTP Status Codes - Delete a Queue Object Value using CDMI Content Type 149
Table 101 – System-Wide Capabilities..... 151
Table 102 – Capabilities for Storage System Metadata 154
Table 103 – Capabilities for Data System Metadata 154
Table 104 – Capabilities for Data Objects 157
Table 105 – Capabilities for Containers 157
Table 106 – Capabilities for Domain Objects 159
Table 107 – Capabilities for Queue Objects..... 160
Table 108 – Request Headers - Read a Capabilities Object using CDMI Content Type 161
Table 109 – Response Headers - Read a Capabilities Object using CDMI Content Type..... 161
Table 110 – Response Message Body - Read a Capabilities Object using CDMI Content Type 162
Table 111 – HTTP Status Codes - Read a Capabilities Object using CDMI Content Type 162
Table 112 – ACE Types 180
Table 113 – Who Identifiers..... 181
Table 114 – ACE Flags 181

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

Table 115 – ACE Bit Masks 182
Table 116 – Storage System Metadata 187
Table 117 – Data System Metadata 188
Table 118 – Provided Values of Data Systems Metadata Items 193
Table 119 – Query Matching Expressions 200
Table 120 – Required Metadata for a Logging Queue 208
Table 121 – Logging Status Metadata 209
Table 122 – Required Metadata for a Notification Queue 211
Table 123 – Notification Status Metadata 214
Table 124 – Required Metadata for a Query Queue 215
Table 125 – Query Status Metadata 216

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

INFORMATION TECHNOLOGY – CLOUD DATA MANAGEMENT INTERFACE (CDMI™)

Foreword

ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards. Their preparation is entrusted to technical committees; any ISO and IEC member body interested in the subject dealt with may participate in this preparatory work. International governmental and non-governmental organizations liaising with ISO and IEC also participate in this preparation.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

The formal decisions or agreements of IEC and ISO on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC and ISO member bodies.

IEC, ISO and ISO/IEC publications have the form of recommendations for international use and are accepted by IEC and ISO member bodies in that sense. While all reasonable efforts are made to ensure that the technical content of IEC, ISO and ISO/IEC publications is accurate, IEC or ISO cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

In order to promote international uniformity, IEC and ISO member bodies undertake to apply IEC, ISO and ISO/IEC publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any ISO/IEC publication and the corresponding national or regional publication should be clearly indicated in the latter.

ISO and IEC provide no marking procedure to indicate their approval and cannot be rendered responsible for any equipment declared to be in conformity with an ISO/IEC publication.

All users should ensure that they have the latest edition of this publication.

No liability shall attach to IEC or ISO or its directors, employees, servants or agents including individual experts and members of their technical committees and IEC or ISO member bodies for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication of, use of, or reliance upon, this ISO/IEC publication or any other IEC, ISO or ISO/IEC publications.

Attention is drawn to the normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 6.

IMPORTANT – The ‘colour inside’ logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its content. Users should therefore print this document using a colour printer.

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

Introduction

This CDMI™ international standard is intended for application developers who are implementing or using cloud storage. It documents how to access cloud storage and to manage the data stored there.

This document is organized as follows:

1 - Scope	Defines the scope of this document
2 - References	Lists the normative references for this document
3 - Terms	Provides terminology used in this document
4 - Conventions	Describes the conventions used in presenting the interfaces and the typographical conventions used in this document
5 - Overview of Cloud Storage	Provides a brief overview of cloud storage and details the philosophy behind this International Standard as a model for the operations
6 - Common Operations	Gives an example of the resources that may be accessed and the representations used to modify them
7 - Interface Standard	Provides a description of HTTP status codes, Cloud Data Management Interface (CDMI) object types, object references, and object manipulations
8 - Data Object Resource Operations	Provides the normative standard of data object resource operations
9 - Container Object Resource Operations	Provides the normative standard of container object resource operations
10 - Domain Object Resource Operations	Provides the normative standard of domain object resource operations
11 - Queue Object Resource Operations	Provides the normative standard of queue object resource operations
12 - Capability Object Resource Operations	Provides the normative standard of capability object resource operations
13 - Exported Protocols	Discusses how virtual machines in the cloud computing environment may use the exported protocols from CDMI containers
14 - Snapshots	Discusses how snapshots are accessed under CDMI containers
15 - Serialization/Deserialization	Discusses serialization and deserialization, including import and export of serialized data under CDMI
16 - Metadata	Provides the normative standard of the metadata used in the interface
17 - Retention and Hold Management	Describes the optional retention management disciplines to be implemented into the system management functions
18 - Scope Specification	Describes the structure of the scope specification for JSON objects
19 - Results Specification	Provides a standardized mechanism to define subsets of CDMI object contents
20 - Logging	Describes CDMI functional logging for object functions, security events, data management events, and queues

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

21 - Notification Queues	Describes how CDMI clients may efficiently discover what changes have occurred to the system
22 - Query Queues	Describes how CDMI clients may efficiently discover what content matches a given set of metadata query criteria or full-content search criteria
Annex A - (normative) Transport Security	Provides normative text for securing the HTTP communications protocol for transferring CDMI messages
Annex B - (informative) Bibliography	Provides informative references that may contain additional useful information

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

1 Scope

This CDMI™ international standard specifies the interface to access cloud storage and to manage the data stored therein. This international standard applies to developers who are implementing or using cloud storage.

2 Normative References

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

The provisions of the referenced specifications other than ISO/IEC, IEC, ISO and ITU documents, as identified in this clause, are valid within the context of this international standard. The reference to such a specification within this international standard does not give it any further status within ISO/IEC. In particular, it does not give the referenced specifications the status of an international standard.

ISO 3166, *Codes for the representation of names of countries and their subdivisions (Parts 1, 2 and 3)*

ISO 4217:2008, *Codes for the representation of currencies and funds*

ISO 8601:2004, *Data elements and interchange formats – Information interchange – Representation of dates and times*

ISO/IEC 9594-8:2008, *Information technology -- Open Systems Interconnection -- The Directory: Public-key and attribute certificate frameworks*

ISO/IEC 14776-414, *SCSI Architecture Model - 4 (SAM-4)*

IEEE Std 1003.1, 2004*, POSIX ERE, The Open Group, Base Specifications Issue 6* - http://www.unix.org/version3/ieee_std.html

RFC 2045, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies* - http://www.ietf.org/rfc/rfc2045.txt

RFC 2119, *Key Words for Use in RFCs to Indicate Requirement Levels* - http://tools.ietf.org/html/rfc2119

RFC 2246, *The TLS Protocol Version 1.0* - http://www.ietf.org/rfc/rfc2246.txt

RFC 2578, *Structure of Management Information Version 2 (SMIv2)* - http://www.ietf.org/rfc/rfc2578.txt

RFC 2616, *Hypertext Transfer Protocol -- HTTP/1.1* - http://www.ietf.org/rfc/rfc2616.txt

RFC 2617, *HTTP Authentication: Basic and Digest Access Authentication* - http://www.ietf.org/rfc/rfc2617.txt

RFC 3280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* - http://www.ietf.org/rfc/rfc3280.txt

RFC 3530, *Network File System (NFS) Version 4 Protocol* - http://www.ietf.org/rfc/rfc3530.txt

RFC 3720, *Internet Small Computer Systems Interface (iSCSI)* - http://www.ietf.org/rfc/rfc3720.txt

RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax* - http://www.ietf.org/rfc/rfc3986.txt

RFC 4346, *The Transport Layer Security (TLS) Protocol Version 1.1* - http://www.ietf.org/rfc/rfc4346.txt

RFC 4627, *The Application/JSON Media Type for JavaScript Object Notation (JSON)* - http://www.ietf.org/rfc/rfc4627.txt

RFC 4648, *The Base16, Base32, and Base64 Data Encodings,* http://www.ietf.org/rfc/rfc4648.txt

RFC 4918, *HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)* - http://www.ietf.org/rfc/rfc4918.txt

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2* - http://www.ietf.org/rfc/rfc5246.txt

RFC 6208, *Cloud Data Management Interface (CDMI) Media Types* - http://www.ietf.org/rfc/rfc6208.txt

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

3 Terms

For the purposes of this document, the following terms and definitions apply.

3.1
Access Control List
ACL
a persistent list, commonly composed of Access Control Entries (ACEs), that enumerates the rights of principals (users and groups) to access resources

3.2
CDMI™
Cloud Data Management Interface

3.3
CIFS
Common Internet File System

3.4
cloud storage
see Data storage as a Service

3.5
CRC
cyclic redundancy check

3.6
CRUD
create, retrieve, update, delete

3.7
Data storage as a Service
DaaS
delivery of virtualized storage and data services on demand over a network, based on a request for a given service level that hides limits to scalability, is either self-provisioned or provisionless, and is billed based on consumption

3.8
domain
a shared user authorization database that contains users, groups, and their security policies and associated accounting information

Note: Each CDMI object belongs to a single domain, and each domain provides user mapping and accounting information.

3.9 eventual consistency
a behavior of transactional systems that does not provide immediate consistency guarantees to provide enhanced system availability and tolerance to network partitioning

3.10
HTTP
HyperText Transfer Protocol

3.11
Infrastructure as a Service
IaaS
delivery over a network of an appropriately configured virtual computing environment, based on a request for a given service level

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

Note: Typically, IaaS is either self-provisioned or provisionless and is billed based on consumption.

3.12
iSCSI
Internet Small Computer Systems Interface (see RFC 3720)

3.13
LUN
Logical Unit Number (see ISO/IEC 14776-414)

3.14
MIME
Multipurpose Internet Mail Extensions (see RFC 2045)

3.15
NFS
Network File System (see RFC 3530)

3.16
object
an entity that has an object ID, a unique URI, and contains state

Note: Types of CDMI objects include data objects, containers, capabilities, domains, and queues.

3.17
object identifier
a globally-unique value assigned at creation time to identify an object

3.18
OCCI
Open Cloud Computing Interface (see OCCI specification)

3.19
Platform as a Service
PaaS
delivery over a network of a virtualized programming environment, consisting of an application deployment stack based on a virtual computing environment

Note: Typically, PaaS is based on IaaS, is either self-provisioned or provisionless, and is billed based on consumption.

3.20
POSIX
Portable Operating System Interface (see IEEE Std 1003.1)

3.21
private cloud
delivery of SaaS, PaaS, IaaS, and/or DaaS to a restricted set of customers, usually within a single organization

Note: Private clouds are created due to issues of trust.

3.22
public cloud
delivery of SaaS, PaaS, IaaS, and/or DaaS to, in principle, a relatively unrestricted set of customers

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

3.23
Representational State Transfer
REST
specific set of principles for defining, addressing, and interacting with resources addressable by URIs (see REST thesis)

3.24
RPO
recovery point objective

3.25
RTO
recovery time objective

3.26
service level
performance targets for a service

3.27
Software as a Service
SaaS
delivery over a network, on demand, of the use of an application

3.28 thin provisioning
technology that allocates the physical capacity of a volume or file system as applications write data, rather than pre-allocating all the physical capacity at the time of provisioning

3.29
Uniform Resource Identifier
URI
compact sequence of characters that identifies an abstract or physical resource (see RFC 3986)

3.30
virtualization
presentation of resources as if they are physical, when in fact, they are decoupled from the underlying physical resources

3.31
WebDAV
Web Distributed Authoring and Versioning (see RFC 4918)

3.32
XAM
eXtensible Access Method (see INCITS 464-2010)

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

4 Conventions

4.1 Interface Format

Each interface description has nine components, as described in Table 1.

Table 1 - Interface Format

Component	Description
Synopsis	The GET, PUT, POST, and DELETE semantics
Delayed Completion of Create	For long-running operations, a description of behavior when the operation does not immediately complete
Capabilities	A description of the supported operations
Request Headers	The request headers, such as Accept, Authorization, Content-Length, Content-Type, X-CDMI-Specification-Version
Request Message Body	A description of the message body contents
Response Headers	The response headers, such as Content-Length, Content-Type
Response Message Body	A description of the message body contents
Response Status	A list of HTTP status codes
Example	An example of the operation

4.2 Typographical Conventions

All code text is shown in a fixed-width font, as follows:

EXAMPLE

```
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2

{
    "mimetype" : "text/plain",
    "metadata" : {

    },
    "value" : "This is the Value of this Data Object"
}
```

4.3 Request and Response Body Requirements

In request and response message body tables, the Requirement column contains one of the following three values:

- Mandatory. The value specified in this row shall be provided.
- Conditional. If the condition(s) specified in the "Description" cell of this row (to the left of the Requirement) is met, the value specified in this row shall be provided. Otherwise it may be provided unless the Description specifically prohibits it, in which case it shall not be provided.
- Optional. The value specified in this row may be provided.

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

4.4 Key Word Requirements

In this international standard, the key words in Table 2 shall be interpreted as described in RFC 2119.

Table 2 - Key Word Requirements

Key Words	Description
shall must required	An action described with any of these key words is unconditionally required.
shall not must not	An action described with either of these key word phrases is unconditionally prohibited.
should recommended	Valid reasons may exist in specific circumstances to ignore a particular action described with either of these key words, but the full implications must be understood and carefully weighed before choosing a different course.
should not not recommended	Valid reasons may exist in specific circumstances to accept a particular action described by either of these key word phrases, but the full implications should be understood and the case carefully weighed before implementing any action described with these key words.
may optional	An action described with either of these key words is truly optional. One vendor may choose to include the option because a particular marketplace requires it or because the vendor feels that it enhances the product, while another vendor may omit the same option. An implementation which does not include a particular option must be prepared to interoperate with another implementation that does include the option, though perhaps with reduced functionality. Likewise, an implementation which does include a particular option must be prepared to interoperate with another implementation that does not include the option (except, of course, for the feature the option provides).

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

5 Overview of Cloud Storage

5.1 Introduction

When discussing cloud storage and standards, it is important to distinguish the various resources that are being offered as services. These resources are exposed to clients as functional interfaces (i.e., data paths) and are managed by management interfaces (i.e., control paths). This international standard explores the various types of interfaces that are part of offerings today and shows how they are related. This international standard defines a model for the interfaces that may be mapped to the various offerings and a model that forms the basis for cloud storage interfaces into the future.

Another important concept in this international standard is that of metadata. When managing large amounts of data with differing requirements, metadata is a convenient mechanism to express those requirements in such a way that underlying data services may differentiate their treatment of the data to meet those requirements.

The appeal of cloud storage is due to some of the same attributes that define other cloud services: pay as you go, the illusion of infinite capacity (elasticity), and the simplicity of use/management. It is therefore important that any interface for cloud storage support these attributes, while allowing for a multitude of business use cases.

5.2 What is Cloud Storage?

The use of the term "cloud" in describing these new models arose from architecture drawings that typically used a cloud as the icon for a network. The cloud represents any-to-any network connectivity in an abstract way. In this abstraction, the network connectivity in the cloud is represented without concern for how it is made to happen.

The cloud abstraction of complexity produces a simple base on which other features can be built. The general cloud model extends this base by adding a pool of resources. An important part of the cloud model is the concept of a pool of resources that is drawn from, on demand, in small increments. A relatively recent innovation that has made this possible is virtualization.

Thus, cloud storage is simply the delivery of virtualized storage on demand. The formal term that is used for this is Data storage as a Service (DaaS).

5.3 Data Storage as a Service

By abstracting data storage behind a set of service interfaces and delivering it on demand, a wide range of actual offerings and implementations are possible. The only type of storage that is excluded from this definition is that which is delivered in fixed-capacity increments instead of based on demand.

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 17826:2012

An important part of any DaaS offering is the support of legacy clients. Support is accommodated with existing standard protocols such as iSCSI (and others) for block and CIFS/NFS or WebDAV for file network storage, as shown in Figure 1.

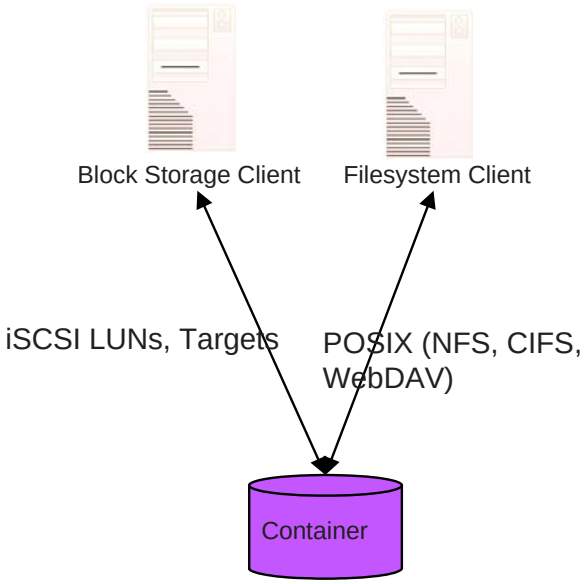

Figure 1 - Existing Data Storage Interface Standards

The difference between the purchase of a dedicated appliance and that of cloud storage is not the functional interface, but the fact that the storage is delivered on demand. The customer pays for either what they actually use or what they have allocated for use. In the case of block storage, a Logical Unit Number (LUN), or virtual volume, is the granularity of allocation. For file protocols, a file system is the unit of granularity. In either case, the actual storage space may be thin provisioned and billed for based on actual usage. Data services, such as compression and deduplication, may be used to further reduce the actual space consumed.

Managing this storage is typically done out of band for these standard data storage interfaces, either through an API, or more commonly, through an administrative browser-based user interface. This out-of-band interface may be used to invoke other data services as well (e.g., snapshot and cloning).

In this model, the underlying storage space exposed by the out-of-band interfaces is abstracted and exposed using the notion of a container. A container is not only a useful abstraction for storage space, but also serves as a grouping of the data stored in it and a point of control for applying data services in the aggregate.

Each data object is created, retrieved, updated, and deleted as a separate resource. In this type of interface, a container, if used, is a simple grouping of data objects for convenience. Nothing prevents the

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

concept of containers from being hierarchical, although any given implementation might support only a single level (see Figure 2).

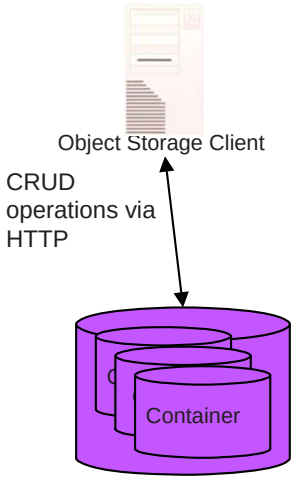

Figure 2 - Storage Interfaces for Object Storage Client Data

5.4 Data Management for Cloud Storage

Many of the initial offerings of cloud storage focused on a kind of "best effort" quality of storage service and ignored most other types of data services. To address the needs of enterprise applications with cloud storage, however, there is an increasing need to offer better quality of service and the deployment of additional data services.

Cloud storage may lose its abstraction and simplicity benefits if new data services that require complex management are added. Cloud storage customers are likely to resist new demands on their time (e.g., setting up backup schedules through dedicated interfaces, deploying data services individually for data elements).

By supporting metadata in a cloud storage interface and prescribing how the storage system and data system metadata is interpreted to meet the requirements of the data, the simplicity required by the cloud storage model may be maintained while still addressing the requirements of enterprise applications and their data.

User metadata is retained by the cloud and may be used to find the data objects and containers by performing a query for specific metadata values. The schema for this metadata may be determined by each application, domain, or user. For more information on support for user metadata, see 16.2.

Storage system metadata is produced/interpreted by the cloud offering and basic storage functions (e.g., modification and access statistics, access control). For more information on support for storage system metadata, see 16.3.

Data system metadata is interpreted by the cloud offering as data requirements that control the operation of underlying data services for that data. It may apply to an aggregation of data objects in a container or to individual data objects, if the offering supports this level of granularity. For more information on support for data system metadata, see 16.4.

5.5 Data and Container Management

There is no reason that managing data and managing containers should involve different interfaces. Therefore, the use of metadata is extended from applying to individual data elements to applying to containers of data as well. Thus, any data placed into a container inherits the data system metadata of the container into which it was placed. When creating a new container within an existing container, the new container would similarly inherit the metadata settings of its parent's data system metadata. After a data

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

element is created, the data system metadata may be overridden at the container or individual data element level, as desired.

Even if the provided interface does not support setting metadata on individual data elements, metadata may still be applied to the containers. In such a case, the interface does not provide a mechanism to override metadata that an individual data element inherits from its parent container. For file-based interfaces that support extended attributes (e.g., CIFS, NFSv4), these extended attributes may be used to specify the data system metadata to override that specified for the container.

5.6 Reference Model for Cloud Storage Interfaces

The Cloud Storage Reference Model is shown in Figure 3.

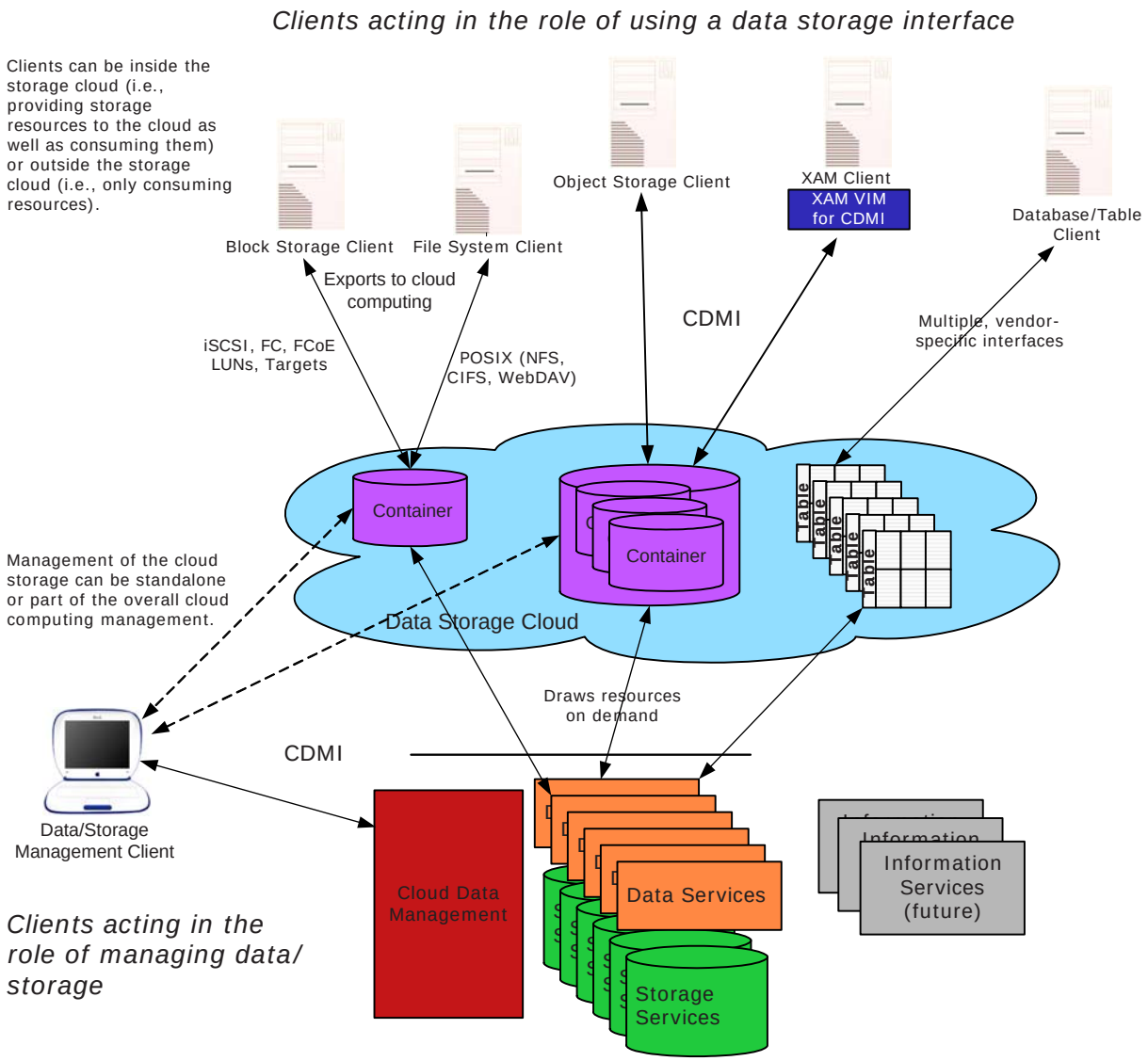

Figure 3 - Cloud Storage Reference Model

This model shows multiple types of cloud data storage interfaces that are able to support both legacy and new applications. All of the interfaces allow storage to be provided on demand, drawn from a pool of resources. The storage capacity is drawn from a pool of storage capacity provided by storage services. The data services are applied to individual data elements, as determined by the data system metadata. Metadata specifies the data requirements on the basis of individual data elements or on groups of data elements (containers).

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

5.7 Cloud Data Management Interface

The Cloud Data Management Interface (CDMI™) shown in Figure 3 may be used to create, retrieve, update, and delete objects in a cloud. The features of the CDMI include functions that

- allow clients to discover the capabilities available in the cloud storage offering;
- manage containers and the data that is placed in them; and
- allow metadata to be associated with containers and the objects they contain.

This international standard divides operations into two types: those that use a CDMI content type in the HTTP body and those that do not. While much of the same data is available via both types, providing both allows for CDMI-aware clients and non-CDMI-aware clients to interact with a CDMI provider.

CDMI may also be used by administrative and management applications to manage containers, domains, security access, and monitoring/billing information, even for storage that is functionally accessible by legacy or proprietary protocols. The capabilities of the underlying storage and data services are exposed so that clients may understand the offering.

Conformant cloud offerings may support a subset of the CDMI, as long as they expose the limitations in the capabilities reported via the interface.

This international standard uses RESTful principles in the interface design where possible (see REST).

CDMI defines both a means to manage the data as well as a means to store and retrieve the data. The means by which the storage and retrieval of data is achieved is termed a data path. The means by which the data is managed is termed the control path. CDMI specifies both a data path and control path interface.

CDMI does not need to be used as the only data path and is able to manage cloud storage properties for any data path interface (e.g., standardized or vendor specific).

Container metadata is used to configure the data requirements of the storage provided through the exported protocol (e.g., block protocol or file protocol) that the container exposes. When an implementation is based on an underlying file system to store data for a block protocol (e.g., iSCSI), the CDMI container provides a useful abstraction for representing the data system metadata for the data and the structures that govern the exported protocols.

A cloud offering may also support domains that allow administrative ownership to be associated with stored objects. Domains allow the standard to (among other things):

- determine how user credentials are mapped to principals used in an Access Control List (ACL),
- allow granting of special cloud-related privileges, and
- allow delegation to external user authorization systems (e.g., LDAP or Active Directory).

Domains may also be hierarchical, allowing for corporate domains with multiple children domains for departments or individuals. The domain concept is also used to aggregate usage data that is used to bill, meter, and monitor cloud use.

Finally, capabilities allow a client to discover the capabilities of a CDMI implementation. Requirements throughout this international standard shall be understood in the context of CDMI capabilities. Mandatory requirements on functionality that is conditioned on a CDMI capability shall not be interpreted to require implementation of that capability, but rather shall be interpreted to apply only to implementations that support the functionality required by that capability.

For example, in 5.10, this international standard states, "Every cloud storage system shall allow object ID-based access to stored objects". This requirement shall be understood in the context that access by object ID is predicated on the presence of the cdmi_object_access_by_ID capability.

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

5.8 Object Model for CDMI

The model for CDMI is shown in Figure 4.

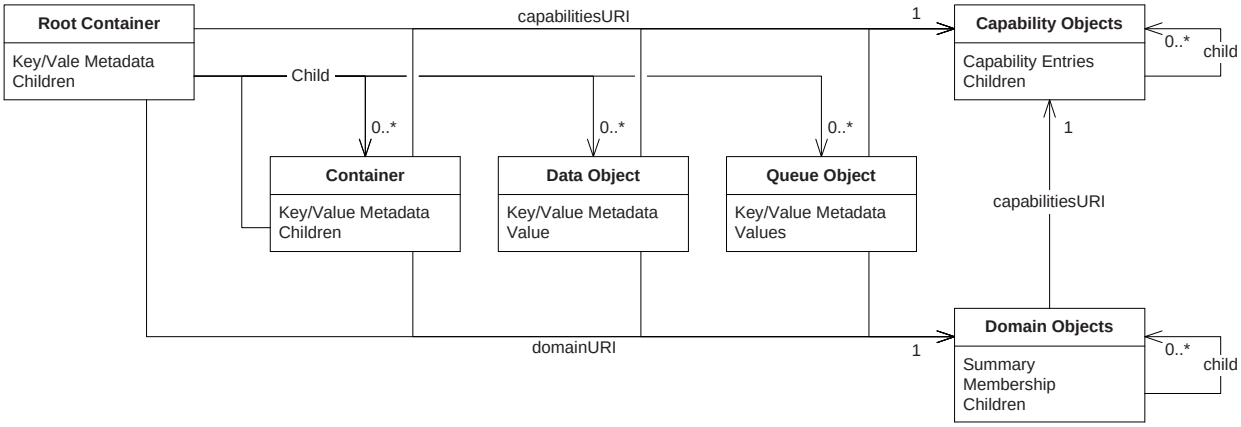

Figure 4 - CDMI Object Model

The five types of resources defined are shown in Table 3. The content type in any given operation is specific to each type of resource.

Table 3 - Types of Resources in the Model

Resource Type	Description	Reference
Data objects	Data objects are used to store values and provide functionality similar to files in a file system.	See Clause 8.
Container objects	Container objects have zero or more children, but do not store values. They provide functionality similar to directories in a file system.	See Clause 9.
Domain objects	Domain objects represent administrative groupings for user authentication and accounting purposes.	See Clause 10.
Queue objects	Queue objects store zero or move values and are accessed in a first-in-first-out manner.	See Clause 11.
Capability objects	Capability objects describe the functionality implemented by a CDMI server and are used by a client to discover supported functionality.	See Clause 12.

For data storage operations, the client of the interface only needs to know about container objects and data objects. All data path implementations are required to support at least one level of containers (see 5.5). Using the CDMI object model (see Figure 4), the client may send a PUT via CDMI (see 5.6) to the new container URI and create a new container with the specified name. Container metadata are optional and are expressed as a series of name-value pairs. After a container is created, a client may send a PUT to create a data object within the newly created container. A subsequent GET will fetch the data object, including the value field.

Queue objects are also defined (see Figure 4) and have special properties for in-order, first in, first-out creation and fetching of queue values. More information on queues may be found in Clause 11.

CDMI defines two namespaces that can be used to access stored objects, a flat object ID namespace and a hierarchical path-based namespace. Support for objects accessed by object ID is indicated by the system-wide capability cdmi_object_access_by_ID, and support for objects accessed by hierarchical path is indicated by the container capability cdmi_create_dataobject found on the root container (and any subcontainers).

Objects are created by ID by performing an HTTP POST against a special URI, designated as /cdmi_objectid/ (see 9.8). Subsequent to creation, objects are modified by performing PUTs using the

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

object ID assigned by the CDMI server, using the /cdmi_objectid/ URI (see 8.6). The same URI is used to retrieve and delete objects by ID.

Objects are created by name by performing an HTTP PUT to the desired path URI (see 8.2). Subsequent to creation, objects are modified by performing PUTs using the object path specified by the client (see 8.6). The same URI is used to retrieve and delete objects by path.

CDMI defines mechanisms so that objects having only an object ID can be assigned a path location within the hierarchical namespace, and so that objects having both an object ID and path can have their path dropped, such that the object only has an object ID. This function is accomplished by using a move modifier to a PUT or POST operation, as shown in Figure 5.

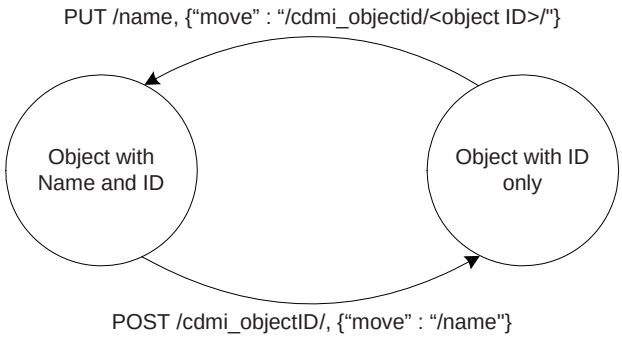

Figure 5 - Object Transitions between Named and ID-only

5.9 CDMI Metadata

CDMI uses many different types of metadata, including HTTP metadata, data system metadata, user metadata, and storage system metadata.

HTTP metadata is metadata that is related to the use of the HTTP protocol (e.g., Content-Length, Content-Type, etc.). HTTP metadata is not specifically related to this international standard but needs to be discussed to explain how CDMI uses the HTTP standard.

CDMI data system metadata, user metadata, and storage system metadata is defined in the form of name-value pairs. Vendor-defined data system metadata and storage system metadata names shall begin with the reverse domain name of the vendor.

Data system metadata is metadata that is specified by a CDMI client and is a component of objects. Data system metadata abstractly specifies the data requirements associated with data services that are deployed in the cloud storage system.

User metadata consists of client-defined JSON strings, arrays, and objects that are stored in the metadata field. The namespace used for user metadata names is self-administered (e.g., using the reverse domain name), and user metadata names shall not begin with the prefix "cdmi_".

Storage system metadata is metadata that is generated by the storage services in the system (e.g., creation time, size) to provide useful information to a CDMI client.

The matrix of the creation and consumption of storage system metadata is shown in Table 4.

Table 4 - Creation/Consumption of Storage System Metadata

	Created by User	**Created By System**
Consumed by User	User metadata	Storage system metadata
Consumed by System	Data system metadata	N/A

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

5.10 Object ID

Every object stored within a CDMI-compliant system shall have a globally unique object identifier (ID) assigned at creation time. The CDMI object ID is a string with requirements for how it is generated and how it obtains its uniqueness. Each offering that implements CDMI is able to produce these identifiers without conflicting with other offerings.

Every cloud storage system shall allow object ID-based access to stored objects by allowing the object's ID to be appended to the root container URI. If the data object "MyDataObject.txt" has an object ID of "00006FFD001001CCE3B2B4F602032653", the following pair of URIs access the same data object:

http://cloud.example.com/root/MyDataObject.txt

http://cloud.example.com/root/cdmi_objectid/00006FFD001001CCE3B2B4F602032653

If containers are supported, they shall also be accessible by object ID. If the container "MyContainer" has an object ID of "00006FFD0010AA33D8CEF9711E0835CA", the following pairs of URIs access the same data object:

http://cloud.example.com/MyContainer/

http://cloud.example.com/cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA/

http://cloud.example.com/MyContainer/MyDataObject.txt

http://cloud.example.com/cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA/MyDataObject.txt

5.11 CDMI Object ID Format

The offering shall create the object ID, which identifies an object. The object ID shall be globally unique and shall conform to the format defined in Figure 6. The native format of an object ID is a variable-length byte sequence and shall be a maximum length of 40 bytes. An application should treat object IDs as opaque byte strings. However, the object ID format is defined such that its integrity may be validated, and independent offerings may assign unique object ID values independently.

0	1	2	3	4	5	6	7	8	9	10	...	38	39
Reserved (zero)	Enterprise Number			Reserved (zero)	Length	CRC		Opaque Data					

Figure 6 - Object ID Format

The fields shown in Figure 6 are defined as follows:

- The reserved bytes shall be set to zero.
- The Enterprise Number field shall be the SNMP enterprise number of the offering organization that created the object ID, in network byte order. See RFC 2578 and http://www.iana.org/assignments/enterprise-numbers. 0 is a reserved value.
- The byte at offset 5 shall contain the full length of the object ID, in bytes.
- The CRC field shall contain a 2-byte (16-bit) CRC in network byte order. The CRC field enables the object ID to be validated for integrity. The CRC field shall be generated by running the

©ISO/IEC 2012 - All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

algorithm (see CRC) across all bytes of the object ID, as defined by the Length field, with the CRC field set to zero. The CRC function shall have the following fields:

— Name : "CRC-16",
— Width : 16,
— Poly : 0x8005,
— Init : 0x0000,
— RefIn : True,
— RefOut : True,
— XorOut : 0x0000, and
— Check : 0xBB3D.

This function defines a 16-bit CRC with polynomial 0x8005, reflected input, and reflected output. This CRC-16 is specified in CRC.

- Opaque data in each object ID shall be unique for a given Enterprise Number.

The native format for an object ID is binary. When necessary, such as when included in URIs and JSON strings, the object ID textual representation shall be encoded using base 16 encoding rules described in RFC 4648 and shall be case insensitive.

5.12 Security

Security, in the context of CDMI, refers to the protective measures employed in managing and accessing data and storage. The specific objectives to be addressed by security include:

- provide a mechanism that assures that the communications between a CDMI client and server may not be read or modified by a third party;
- provide a mechanism that allows CDMI clients and servers to provide an assurance of their identity;
- provide a mechanism that allows control of the actions a CDMI client is permitted to perform on a CDMI server;
- provide a mechanism for records to be generated for actions performed by a CDMI client on a CDMI server;
- provide mechanisms to protect data at rest;
- provide a mechanism to eliminate data in a controlled manner; and
- provide mechanisms to discover the security capabilities of a particular implementation.

Security measures within CDMI may be summarized as

- transport security,
- user and entity authentication,
- authorization and access controls,
- data integrity,
- data and media sanitization,
- data retention,
- protections against malware,
- data at-rest encryption, and
- security capabilities.

With the exception of both the transport security and the security capabilities, which are mandatory to implement, the security measures may vary significantly from implementation to implementation.

When security is a concern, the CDMI client should begin with a series of security capability lookups (see 12.1.1) to determine the exact nature of the security features that are available. Based on the values of

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

these capabilities, a risk-based decision should be made as to whether the CDMI server should be used. This is particularly true when the data to be stored in the cloud storage is sensitive or regulated in a way that requires stored data to be protected (e.g., encrypted) or handled in a particular manner (e.g., full accountability and traceability of management and access).

HTTP is the mandatory transport mechanism, and HTTP over TLS (i.e., HTTPS) is the mechanism used to secure the communications between CDMI clients and servers. To ensure both security and interoperability, all CDMI implementations shall implement the Transport Layer Security (TLS) protocol as described in Annex A, but its use by CDMI clients and servers is optional.

5.13 Required HTTP Support

5.13.1 RFC 2616 Support Requirements

A conformant implementation of CDMI shall also be a conformant implementation of RFC2616 (see RFC 2616) (i.e., HTTP 1.1). The subclauses below list the sections of RFC 2616 that shall be supported; however, this list is not comprehensive.

5.13.2 Content-Type Negotiation

For CDMI operations, media types for CDMI objects are used, as defined in RFC 6208.

A client may optionally supply an HTTP Accept header, as per section 14.1 of RFC 2616. If a client is restricting the response to a specific CDMI media type, the corresponding media type shall be specified in the Accept header. Otherwise, the Accept header may contain "*/*" or a list of media types, or it may be omitted.

If a request message body is present, the client shall include a Content-Type header, as per section 14.17 of RFC 2616. If the client does not provide a Content-Type header when required or provides a media type in the Content-Type header that does not match with the existing resource media type, the server shall return an HTTP status code of 400 Bad Request.

If a response message body is present, the server shall provide a Content-Type header.

This international standard may further qualify content negotiation (e.g., in 9.3, the absence of a Content-Type header has a specific meaning).

5.13.3 Range Support

The server shall support HTTP Range headers and partial content responses (see Section 14.16 of RFC 2616).

5.13.4 URI Escaping

Percent escaping of reserved characters specified in RFC 3986 shall be applied to all text strings used in URIs. This includes user-supplied field names, metadata names, object names, container names and domain names when used in URIs.

Field names and values shall not be escaped when stored and sent in request and response message bodies.

EXAMPLE A client retrieving a metadata item named "@user" from a container object with the name of "@MyContainer" would perform the following request:

```
GET /%40MyContainer/?objectName;metadata:%40user HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
```


©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

The response shall be:

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2

{
    "objectName": "@MyContainer",
    "metadata": {
        "@user": "test"
    }
}
```

5.13.5 Use of URIs

The format and syntax of URIs are defined by RFC 3986.

Every CDMI client shall maintain one or more root URIs that each correspond to a root container on the CDMI server. Since all URIs to CDMI containers end in a trailing slash, all root URIs will end in a trailing slash.

All URIs in this international standard are relative to the root URI unless otherwise noted. As a consequence, the algorithm used for calculating the resolved URI is as described in Section 5.2 of RFC 3986.

Table 5 shows how relative URIs are resolved against root URIs.

Table 5 - Relative URIs Resolved Against Root URIs

Root URI	+ Relative URI	=> Resolved URI
http://cloud.example.com/	cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject
http://cloud.example.com/	/cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject
http://cloud.example.com/p1/	cdmi_object/testObject	http://cloud.example.com/p1/cdmi_object/testObject
http://cloud.example.com/p1/	/cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject
http://cloud.example.com/p1/p2/	cdmi_object/testObject	http://cloud.example.com/p1/p2/cdmi_object/ testObject
http://cloud.example.com/p1/p2/	/cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject

This international standard places no restrictions on root and relative URIs. All of the examples in Table 5 are valid, use a root URI of http://cloud.example.com/, and return absolute path references, as shown in the second line of Table 5.

5.13.6 Reserved Characters

The name of CDMI data objects, container objects, queue objects, domain objects and capability objects shall not contain the "/" or "?" characters, as these characters are reserved for delimiters.

5.14 Time Representations

Unless otherwise specified, all date/time values are in the ISO 8601:2004 extended representation (YYYY-MM-DDThh:mm:ss.ssssssZ). The full precision shall be specified, the sub-second separator shall be a ".", the Z UTC zone indicator shall be included, and all timestamps shall be in UTC time zone. The YYYY-MM-DDT24:00:00.000000Z hour shall not be used, and instead, it shall be represented as YYYY-MM-DDT00:00:00.000000Z.

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

Unless otherwise specified, all date/time intervals are in the ISO 8601:2004 start date/end date representation (YYYY-MM-DDThh:mm:ss.ssssssZ/YYYY-MM-DDThh:mm:ss.ssssssZ). The end-date shall be equal to or later than the start-date. The full precision shall be specified, the sub-second separator shall be a ".", the Z UTC zone indicator shall be included, and all timestamps shall be in UTC time zone. The YYYY-MM-DDT24:00:00.000000Z hour shall not be used, and instead, it shall be represented as YYYY-MM-DDT00:00:00.000000Z.

5.15 Backwards Compatibility

5.15.1 Value Transfer Encoding

CDMI version 1.0.1 introduces the concept of value transfer encoding to enable the storage and retrieval of arbitrary binary data via CDMI content-type operations. Data objects created by CDMI 1.0 clients through CDMI content-type operations shall have a value transfer encoding of "utf-8", and data objects created through non-CDMI content-type operations shall have a value transfer encoding of "base64".

Data objects with a value transfer encoding of base 64 shall not have their value field accessible to CDMI 1.0 clients through CDMI content-type operations. Attempts to read the value of these objects shall return an empty value field ("") to these clients. CDMI 1.0 clients can detect this condition when the cdmi_size metadata is not 0 and the value field is empty.

5.15.2 Container Export Capabilities

CDMI version 1.0.2 normalizes the names of capabilities used by a client to discover if a container can be exported via various protocols and deprecates the following container export capability names:

- cdmi_cifs_export,
- cdmi_nfs_export,
- cdmi_iscsi_export, and
- cdmi_occi_export.

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

6 Common Operations

6.1 Overview

All examples included in this international standard are informative.

This clause includes examples for the following CDMI content-type operations:

- discovering the capabilities of a cloud storage provider (see 6.2),
- creating a new container (see 6.3),
- creating a new data object (see 6.4),
- listing the contents of a container (see 6.5),
- reading the contents of a data object (see 6.6),
- reading only the value of a data object (see 6.7), and
- deleting a data object (see 6.8).

6.2 Discover the Capabilities of a Cloud Storage Provider

EXAMPLE Perform a GET to the capabilities URI:

```
GET /cdmi_capabilities/ HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-capability
X-CDMI-Specification-Version: 1.0.2
```

The following shows the response.

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-capability
X-CDMI-Specification-Version: 1.0.2

{
    "objectType" : "application/cdmi-capability",
    "objectID" : "00007E7F0010CEC234AD9E3EBFE9531D",
    "objectName" : "cdmi_capabilities/",
    "parentURI" : "/",
    "parentID" : "00007E7F0010DCECC805FB6D195DDBCB",
    "capabilities" : {
        "cdmi_domains" : "true",
        "cdmi_export_nfs" : "true",
        "cdmi_export_webdav" : "true",
        "cdmi_export_iscsi" : "true",
        "cdmi_queues" : "true",
        "cdmi_notification" : "true",
        "cdmi_query" : "true",
        "cdmi_metadata_maxsize" : "4096",
        "cdmi_metadata_maxitems" : "1024",
        "cdmi_size" : "true",
        "cdmi_list_children" : "true",
        "cdmi_read_metadata" : "true",
        "cdmi_modify_metadata" : "true",
        "cdmi_create_container" : "true",
        "cdmi_delete_container" : "true"
    },
    "childrenrange" : "0-3",
    "children" : [
        "domain/",
        "container/",
        "dataobject/",
        "queue/"
    ]
}
```

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

6.3 Create a New Container

EXAMPLE Perform a PUT to the new container URI:

```
PUT /MyContainer/HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2

{
    "metadata" : {

    }
}
```

The following shows the response.

```
HTTP/1.1 201 Created
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2

{
    "objectType" : "application/cdmi-container",
    "objectID" : "00007E7F00102E230ED82694DAA975D2",
    "objectName" : "MyContainer/",
    "parentURI" : "/",
    "parentID" : "00007E7F0010128E42D87EE34F5A6560",
    "domainURI" : "/cdmi_domains/MyDomain/",
    "capabilitiesURI" : "/cdmi_capabilities/container/",
    "completionStatus" : "Complete",
    "metadata" : {
        "cdmi_size" : "0"
    },
    "childrenrange" : "",
    "children" : [

    ]
}
```

6.4 Create a Data Object in a Container

EXAMPLE Perform a PUT to the new data object URI:

```
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2

{
    "mimetype" : "text/plain",
    "metadata" : {

    },
    "value" : "Hello CDMI World!"
}
```

The following shows the response.

```
HTTP/1.1 201 Created
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2

{
    "objectType" : "application/cdmi-object",
    "objectID" : "00007E7F0010BD1CB8FF1823CF05BEE4",
```


©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

```
    "objectName" : "MyDataObject.txt",
    "parentURI" : "/MyContainer/",
    "parentID" : "00007E7F00102E230ED82694DAA975D2",
    "domainURI" : "/cdmi_domains/MyDomain/",
    "capabilitiesURI" : "/cdmi_capabilities/dataobject/",
    "completionStatus" : "Complete",
    "mimetype" : "text/plain",
    "metadata" : {
        "cdmi_size" : "17"
    }
}
```

6.5 List the Contents of a Container

EXAMPLE Perform a GET to the container URI:

```
GET /MyContainer/ HTTP/1.1
Host: cloud.example.com
Accept: */*
X-CDMI-Specification-Version: 1.0.2
```

The following shows the response.

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2

{
    "objectType" : "application/cdmi-container",
    "objectID" : "00007E7F00102E230ED82694DAA975D2",
    "objectName" : "MyContainer/",
    "parentURI" : "/",
    "parentID" : "00007E7F0010128E42D87EE34F5A6560",
    "domainURI" : "/cdmi_domains/MyDomain/",
    "capabilitiesURI" : "/cdmi_capabilities/container/",
    "completionStatus" : "Complete",
    "metadata" : {
        "cdmi_size" : "83"
    },
    "childrenrange" : "0-0",
    "children" : [
        "MyDataObject.txt"
    ]
}
```

6.6 Read the Contents of a Data Object

EXAMPLE GET from the data object URI:

```
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
```

The following shows the response.

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2

{
    "objectType": "application/cdmi-object",
    "objectID": "00007E7F0010BD1CB8FF1823CF05BEE4",
    "objectName": "MyDataObject.txt",
    "parentURI": "/MyContainer/",
```

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

```
        "parentID" : "00007E7F00102E230ED82694DAA975D2",
        "domainURI": "/cdmi_domains/MyDomain/",
        "capabilitiesURI": "/cdmi_capabilities/dataobject/",
        "completionStatus": "Complete",
        "mimetype": "text/plain",
        "metadata": {
            "cdmi_size": "17"
        },
        "valuetransferencoding": "utf-8",
        "valuerange": "0-16",
        "value": "Hello CDMI World!"
    }
```

6.7 Read Only the Value of a Data Object

EXAMPLE Perform a GET to the data object URI:

```
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
```

The following shows the response.

```
HTTP/1.1 200 OK
Content-Type: text/plain

Hello CDMI World!
```

6.8 Delete a Data Object

EXAMPLE Perform a DELETE to the data object URI:

```
DELETE /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
X-CDMI-Specification-Version: 1.0.2
```

The following shows the response.

```
HTTP/1.1 204 No Content
```


©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

7 Interface Standard

7.1 HTTP Status Codes

HTTP status codes (see Table 6) are used to convey the results of the RESTful operations and to follow the basic semantics of HTTP with minimal overloading. Other HTTP status codes are not part of this international standard and retain their original semantics from HTTP 1.1.

Table 6 - HTTP Status Codes

Status Code	HTTP Name	Description
200	OK	The request has succeeded.
201	Created	The resource was created successfully.
202	Accepted	The long-running operation was accepted for processing.
204	No Content	The operation was successful; no data was returned.
302	Found	The resource is a reference to another resource.
400	Bad Request	The request contents are missing or invalid.
401	Unauthorized	The authentication/authorization credentials are invalid.
403	Forbidden	The client lacks the proper authorization to perform this request.
404	Not Found	The resource was not found at the specified URI.
406	Not Acceptable	No content can be produced at this URI that matches the request.
409	Conflict	The operation conflicts with a non-CDMI™ access protocol lock or may cause a state transition error on the server.

7.2 Object References

Object references are URIs within the cloud storage namespace that redirect to another URI within the same or another cloud storage namespace. References are similar to soft links in a file system. The cloud does not guarantee that the referenced URI will be valid after the time of creation.

References are visible as children in a container and are distinguished from non-references by a trailing "?" character added to the reference name. Performing an operation (with the exception of create or delete) to a reference URI will result in a 302 Found HTTP redirect, with the Location HTTP header containing the redirect destination URI that was specified at the time the reference was created. The reference's destination URI shall not be changed after a reference has been created.

To continue, when CDMI clients receive a 302 Found redirect, they should retry the operation using the URI contained within the Location header.

A delete operation on a reference URI shall delete the reference. References cannot be updated. To update the destination of a redirect, the client shall first delete the reference and then create a new reference to the desired destination.

EXAMPLE 1 GET to a URI, where the URI is a reference:

```
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
```

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

The following shows the response.

```
HTTP/1.1 302 Found
Location: http://cloud.example.com/MyContainer/MyOtherDataObject.txt
```

References by object ID shall always redirect to a URI that ends with the same object ID as the request URI.

EXAMPLE 2 GET to an object ID URI, where the URI is a reference:

```
GET /cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
```

The following shows the response.

```
HTTP/1.1 302 Found
Location: http://archive.example.com/cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA
```


©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

8 Data Object Resource Operations

8.1 Overview

Data objects are the fundamental storage component within CDMI™ and are analogous to files within a file system. Each data object has a set of well-defined fields that include:

- a single value; and
- optional metadata that is generated by the cloud storage system and specified by the cloud user.

Data objects are addressed in CDMI in two ways:

- by name (e.g., http://cloud.example.com/dataobject); and
- by object ID (e.g., http://cloud.example.com/cdmi_objectid/ 0000706D0010B84FAD185C425D8B537E).

Every data object has a single, globally-unique object identifier (ID) that remains constant for the life of the object. Each data object shall have one or more URI addresses that allow the object to be accessed.

Every data object has a parent object from which the data object inherits data system metadata that is not explicitly specified in the data object itself. Thus, the "budget.xls" data object stored at the following URI would inherit data system metadata from its parent container, "finance":

http://cloud.example.com/finance/budget.xls

Individual fields within a data object may be accessed by specifying the field name after a question mark "?" that is appended to the end of the data object URI. Thus, the following URI returns the value field in the response message body:

http://cloud.example.com/dataobject?value

The encoding of the data transported in the data object value field depends on the data object valuetransferencoding field.

- If the value transfer encoding of the object is set to "utf-8", the data stored in the value of the data object shall be a valid UTF-8 string and shall be transported as a UTF-8 string in the value field.
- If the value transfer encoding of the object is set to "base64", the data stored in the value of the data object can contain arbitrary binary sequences, and it shall be transported as a base 64-encoded string in the value field.

Specific ranges of the value of a data object may be accessed by specifying a byte range after the value field name. Thus, the following URI returns the first thousand bytes in the value field:

http://cloud.example.com/dataobject?value:0-999

Because a byte range of a UTF-8 string is often not a valid UTF-8 string, the response to a range request shall always be transported in the value field as a base 64-encoded string. Likewise, when updating a range of bytes within the value of a data object, the contents of the value field shall be transported as a base 64-encoded string.

Byte ranges are specified as single inclusive byte ranges as per Section 14.35.1 of RFC 2616.

A list of unique fields, separated by a semicolon ";" may be specified, allowing multiple fields to be accessed in a single request. Thus, the following URI returns the value and metadata fields in the response message body:

http://cloud.example.com/dataobject?value;metadata

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

When a client provides fields that are not defined in this international standard or deserializes an object containing fields that are not defined in this international standard, these fields shall be stored as part of the object but shall not be interpreted.

8.1.1 Data Object Metadata

Data object metadata may also include arbitrary user-supplied metadata and data system metadata, as specified in Clause 16. Metadata shall be stored as a valid UTF-8 string. Binary data stored in user metadata shall be first encoded such that it can be contained in a UTF-8 string, with the use of base 64 encoding recommended.

8.1.2 Data Object Consistency

Writing to a data object is an atomic operation.

- If a client reads a data object simultaneously with a write to that same data object, the reading client shall get either the old version or the new version, but not a mixture of both.
- If a write is terminated due to errors, the contents of the data object shall be as if the write never occurred (i.e., writes are atomic in the face of errors).

The timestamp returned in the response to a write indicates whether the write is the newest (i.e., the write whose data is returned to subsequent reads, until another write is processed). If the timestamp returned for one write shows a more recent time than the timestamp for another write, then the write with the new timestamp shall be the one whose data is currently stored in the data object wherever the two writes overlap.

Range writes can result in a gap in an object value that have had no data written to them. Reading from a gap in a data object value shall return zero for each byte read.

Implementations of this international standard shall provide the atomicity features described in this subclause for data objects that are accessed via CDMI. The atomicity properties of data objects that are accessed by protocols other than CDMI are outside the scope of this international standard.

8.1.3 Data Object Representations

The representations in this clause are shown using JSON notation. Both clients and servers shall support UTF-8 JSON representation. The request and response message body JSON fields may be specified or returned in any order, with the exception that, if present, for data objects, the valuerange and value fields shall appear last and in that order.

8.2 Create a Data Object Using CDMI Content Type

8.2.1 Synopsis

To create a new data object, the following request shall be performed:

```
PUT <root URI>/<ContainerName>/<DataObjectName>
```

To create a new data object by ID, see 9.9.

Where:

- <root URI> is the path to the CDMI cloud.
- <ContainerName> is zero or more intermediate containers that already exist, with one slash (i.e., "/") between each pair of container names.
- <DataObjectName> is the name specified for the data object to be created.

After it is created, the data object shall also be accessible at <root URI>/cdmi_objectid/<objectID>.

©ISO/IEC 2012 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

8.2.2 Delayed Completion of Create

In response to a create operation for a data object, the server may return 202 Accepted to indicate that the object is in the process of being created. This response is useful for long-running operations (e.g., copying a large data object from a source URI). Such a response has the following implications.

- The server shall return a Location header with a URI to the object to be created along with an HTTP status code of 202 Accepted.
- With 202 Accepted, the server implies that the following checks have passed:
  - — user authorization for creating the object;
  - — user authorization for read access to any source object for move, copy, serialize, or deserialize; and
  - — availability of space to create the object or at least enough space to create a URI to report an error.
- A client might not be able to immediately access the created object, e.g., due to delays resulting from the implementation's use of eventual consistency.

The client performs GET operations to the URI to track the progress of the operation. In response, the server returns two fields in its response message body to indicate progress.

- A mandatory completionStatus text field contains either "Processing", "Complete", or an error string starting with the value "Error".
- An optional percentComplete field contains the percentage of the operation that has completed (0 to 100).

GET shall not return any value for the data object when completionStatus is not "Complete". If the final result of the create operation is an error, the URI is created with the completionStatus field set to the error message. It is the client's responsibility to delete the URI after the error has been noted.

8.2.3 Capabilities

The following capabilities describe the supported operations that may be performed when creating a new data object:

- Support for the ability to create a new data object is indicated by the presence of the cdmi_create_dataobject capability in the parent container.
- If the object being created in the parent container is a reference, support for that ability is indicated by the presence of the cdmi_create_reference capability in the parent container.
- If the new data object is a copy of an existing data object, support for the ability to copy is indicated by the presence of the cdmi_copy_dataobject capability in the parent container.
- If the new data object is the destination of a move, support for the ability to move the data object is indicated by the presence of the cdmi_move_dataobject capability in the parent container.
- If the new data object is the destination of a deserialize operation, support for the ability to deserialize the source data object is indicated by the presence of the cdmi_deserialize_dataobject capability in the parent container.
- If the new data object is the destination of a serialize operation, support for the ability to serialize the source data object is indicated by the presence of the cdmi_serialize_dataobject, cdmi_serialize_container, cdmi_serialize_domain, or cdmi_serialize_queue capability in the parent container.

©ISO/IEC 2012 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012

8.2.4 Request Headers

The HTTP request headers for creating a CDMI data object using CDMI content type are shown in Table 7.

Table 7 - Request Headers for Creating a CDMI Data Object using CDMI Content Type

Header	Type	Description	Requirement
Accept	Header String	"application/cdmi-object" or a consistent value as per clause 5.13.2	Optional
Content-Type	Header String	"application/cdmi-object"	Mandatory
X-CDMI-Specification-Version	Header String	A comma-separated list of versions supported by the client, e.g., "1.0.2, 1.5, 2.0"	Mandatory
X-CDMI-Partial	Header String	"true". Indicates that the newly created object is part of a series of writes and the value has not yet been fully populated. If X-CDMI-Partial is present, the completionStatus field in the Response Body shall be set to "Processing".	Optional

8.2.5 Request Message Body

The request message body fields for creating a data object using CDMI content type are shown in Table 8.

Table 8 - Request Message Body - Create a Data Object using CDMI Content Type (Sheet 1 of 3)

Field Name	Type	Description	Requirement
mimetype	JSON String	MIME type of the data contained within the value field of the data object • This field may be included when creating by value or when deserializing, serializing, copying, and moving a data object. • This field shall be stored as part of the object. • If this field is not specified, the value of "text/plain" shall be assigned as the field value. • This field shall not be included when creating a reference. • This MIME type value shall be converted to lower case before being stored.	Optional
[a]Only one of these fields shall be specified in any given operation. Except for value, these fields shall not be stored. If more than one of these fields is supplied, the server shall respond with a 400 Bad Request error response.			

©ISO/IEC 2012 - All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 17826:2012