



ISO/IEC 29341-4-12

Edition 1.0 2008-11

INTERNATIONAL STANDARD

**Information technology – UPnP Device Architecture –
Part 4-12: Audio Video Device Control Protocol –
Level 2 – Content Directory Service**

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008



THIS PUBLICATION IS COPYRIGHT PROTECTED
Copyright © 2008 ISO/IEC, Geneva, Switzerland

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 IEC or IEC's member National Committee in the country of the requester.

If you have any questions about ISO/IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,...). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008



ISO/IEC 29341-4-12

Edition 1.0 2008-11

INTERNATIONAL STANDARD

Information technology – UPnP Device Architecture –
Part 4-12: Audio Video Device Control Protocol –
Level 2 – Content Directory Service

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE

XB

ICS 35.200

ISBN 2-8318-1006-7

CONTENTS

FOREWORD	14
ORIGINAL UPNP DOCUMENTS (informative)	16
1 Overview and Scope	18
1.1 Introduction	18
1.2 Notation	18
1.2.1 Data Types	19
1.2.2 Strings Embedded in Other Strings	19
1.2.3 Extended Backus-Naur Form	19
1.2.3.1 Typographic conventions for EBNF	19
1.3 Derived Data Types	20
1.3.1 Comma Separated Value (CSV) Lists	20
1.4 Management of XML Namespaces in Standardized DCPs	21
1.4.1 Namespace Prefix Requirements	23
1.4.2 Namespace Names, Namespace Versioning and Schema Versioning	24
1.4.3 Namespace Usage Examples	25
1.5 Vendor-defined Extensions	25
1.6 References	26
2 Service Modeling Definitions	29
2.1 Service Type	29
2.2 Terms	29
2.2.1 object	29
2.2.2 property	29
2.2.2.1 Multi-valued property	30
2.2.2.2 Single-valued property	30
2.2.3 class	30
2.2.4 <u>item</u>	30
2.2.5 <u>container</u>	30
2.2.6 container modification	30
2.2.7 XML Document	31
2.2.8 XML Fragment	31
2.2.9 <u>ContainerUpdateID</u>	31
2.2.10 <i>reference, reference item, referenced item</i>	31
2.2.11 CDS feature	32
2.3 State Variables	32
2.3.1 State Variable Overview	33
2.3.2 <u>SearchCapabilities</u>	34
2.3.3 <u>SortCapabilities</u>	34
2.3.4 <u>SortExtensionCapabilities</u>	34
2.3.5 <u>SystemUpdateID</u>	35
2.3.6 <u>ContainerUpdateIDs</u>	35
2.3.7 <u>TransferIDs</u>	36
2.3.8 <u>FeatureList</u>	36
2.3.9 <u>A ARG TYPE ObjectID</u>	37
2.3.10 <u>A ARG TYPE Result</u>	37
2.3.11 <u>A ARG TYPE SearchCriteria</u>	37
2.3.11.1 <u>SearchCriteria</u> String Syntax	37

2.3.11.2	<u>SearchCriteria</u> String Semantics and Examples	38
2.3.12	<u>A ARG TYPE BrowseFlag</u>	39
2.3.13	<u>A ARG TYPE Filter</u>	39
2.3.14	<u>A ARG TYPE SortCriteria</u>	40
2.3.15	<u>A ARG TYPE Index</u>	41
2.3.16	<u>A ARG TYPE Count</u>	41
2.3.17	<u>A ARG TYPE UpdateID</u>	41
2.3.18	<u>A ARG TYPE TransferID</u>	42
2.3.19	<u>A ARG TYPE TransferStatus</u>	42
2.3.20	<u>A ARG TYPE TransferLength</u>	42
2.3.21	<u>A ARG TYPE TransferTotal</u>	42
2.3.22	<u>A ARG TYPE TagValueList</u>	42
2.3.23	<u>A ARG TYPE URI</u>	42
2.4	Eventing and Moderation	43
2.5	Actions	44
2.5.1	<u>GetSearchCapabilities()</u>	45
2.5.1.1	Arguments	45
2.5.1.2	Dependency on State	45
2.5.1.3	Effect on State	45
2.5.1.4	Errors	45
2.5.2	<u>GetSortCapabilities()</u>	45
2.5.2.1	Arguments	45
2.5.2.2	Dependency on State	45
2.5.2.3	Effect on State	45
2.5.2.4	Errors	46
2.5.3	<u>GetSortExtensionCapabilities()</u>	46
2.5.3.1	Arguments	46
2.5.3.2	Dependency on State	46
2.5.3.3	Effect on State	46
2.5.3.4	Errors	46
2.5.4	<u>GetFeatureList()</u>	46
2.5.4.1	Arguments	46
2.5.4.2	Dependency on State	46
2.5.4.3	Effect on State	46
2.5.4.4	Errors	47
2.5.5	<u>GetSystemUpdateID()</u>	47
2.5.5.1	Arguments	47
2.5.5.2	Dependency on State	47
2.5.5.3	Effect on State	47
2.5.5.4	Errors	47
2.5.6	<u>Browse()</u>	47
2.5.6.1	Arguments	47
2.5.6.2	Dependency on State	48
2.5.6.3	Effect on State	48
2.5.6.4	Errors	49
2.5.7	<u>Search()</u>	49
2.5.7.1	Arguments	49
2.5.7.2	Dependency on State	50
2.5.7.3	Effect on State	50

2.5.7.4	Errors	50
2.5.8	<u>CreateObject()</u>	51
2.5.8.1	<u>res</u> Property Creation	51
2.5.8.2	Arguments	54
2.5.8.3	Dependency on State	54
2.5.8.4	Effect on State	54
2.5.8.5	Errors	55
2.5.9	<u>DestroyObject()</u>	55
2.5.9.1	Arguments	56
2.5.9.2	Dependency on State	56
2.5.9.3	Effect on State	56
2.5.9.4	Errors	56
2.5.10	<u>UpdateObject()</u>	56
2.5.10.1	Arguments	58
2.5.10.2	Dependency on State	58
2.5.10.3	Effect on State	58
2.5.10.4	Errors	59
2.5.11	<u>MoveObject()</u>	59
2.5.11.1	Arguments	60
2.5.11.2	Dependency on State	60
2.5.11.3	Effect on State	60
2.5.11.4	Errors	61
2.5.12	<u>ImportResource()</u>	61
2.5.12.1	Arguments	61
2.5.12.2	Dependency on State	62
2.5.12.3	Effect on State	62
2.5.12.4	Errors	62
2.5.13	<u>ExportResource()</u>	62
2.5.13.1	Arguments	62
2.5.13.2	Dependency on State	62
2.5.13.3	Effect on State	63
2.5.13.4	Errors	63
2.5.14	<u>DeleteResource()</u>	63
2.5.14.1	Arguments	63
2.5.14.2	Dependency on State	64
2.5.14.3	Effect on State	64
2.5.14.4	Errors	64
2.5.15	<u>StopTransferResource()</u>	64
2.5.15.1	Arguments	64
2.5.15.2	Dependency on State	64
2.5.15.3	Effect on State	64
2.5.15.4	Errors	64
2.5.16	<u>GetTransferProgress()</u>	65
2.5.16.1	Arguments	65
2.5.16.2	Dependency on State	65
2.5.16.3	Effect on State	65
2.5.16.4	Errors	65

2.5.17	<u>CreateReference()</u>	65
2.5.17.1	Arguments	66
2.5.17.2	Dependency on State	66
2.5.17.3	Effect on State	66
2.5.17.4	Errors	66
2.5.18	Non-Standard Actions Implemented by a UPnP Vendor	66
2.5.19	Common Error Codes	67
2.6	Theory of Operation (Informative)	69
2.6.1	Introduction	69
2.6.2	Content Setup for Browsing and Searching	69
2.6.3	Browsing	69
2.6.3.1	Retrieving Sort Capabilities	70
2.6.3.2	Browsing the Root Level Metadata	70
2.6.3.3	Browsing the Children of the Root Level	71
2.6.3.4	Browsing the Children of the My Music Folder	72
2.6.3.5	Browsing the Children of the Singles Soundtrack Music Album	72
2.6.3.6	Browsing the Children of the Album Art Folder	74
2.6.4	Searching	74
2.6.4.1	Retrieving Search Capabilities	75
2.6.4.2	Search for All Content Created by the performer Sting	75
2.6.4.3	Search for all Photos Taken During the Month of October	76
2.6.4.4	Search for All Objects in the My Photos Folder Containing the Word "Christmas"	77
2.6.4.5	Search for all <u>album</u> objects in the ContentDirectory service	77
2.6.5	Browsing, Searching, and References	79
2.6.5.1	Creating a reference to a photo in the Mexico Trip album inside the Christmas album	79
2.6.5.2	Search for All Photos Taken During the Month of October	79
2.6.5.3	Deletion of the Reference to the Photo in the Mexico Trip Album	80
2.6.6	Object Creation	80
2.6.6.1	Creating a New Object	80
2.6.6.2	Example: Creating a New MusicTrack in the Album1 (<u>@id</u> = 10)	80
2.6.7	Object Resource Binding (Importing a Resource)	81
2.6.7.1	Transfer Using the <u>ImportResource()</u> Action	81
2.6.7.2	Transfer Using Direct HTTP POST	82
2.6.8	Exporting ContentDirectory Resources	82
2.6.8.1	Transfer Using the <u>ExportResource()</u> Action	83
2.6.8.2	Transfer using HTTP GET	84
2.6.9	Playlist Manipulation	84
2.6.9.1	Playlist File Representation in the ContentDirectory Service	84
2.6.9.2	Playlist File Generation	84
2.6.10	Internet Content Representation	85
2.6.11	Bookmark Manipulation	86
2.6.11.1	<u>bookmarkItem</u> Example	86
2.6.11.2	Creating and Destroying Bookmarks	88
2.6.11.3	Browsing Bookmarks	92
2.6.12	Vendor Metadata Extensions	98
3	XML Service Description	99
4	Test	108

Annex A (normative) Schemas	109
A.1 DIDL-Lite	109
A.2 UPnP Elements	109
A.3 Dublin Core Subset Elements	109
A.4 <u>FeatureList</u> State Variable Schema	109
Annex B (normative) AV Working Committee Properties	110
B.1 Base Properties	110
B.1.1 <u>@id</u>	111
B.1.2 <u>@parentID</u>	111
B.1.3 <u>@refID</u>	111
B.1.4 <u>@restricted</u>	111
B.1.5 <u>@searchable</u>	111
B.1.6 <u>@childCount</u>	111
B.1.7 <u>dc:title</u>	112
B.1.8 <u>dc:creator</u>	112
B.1.9 <u>res</u>	112
B.1.10 <u>upnp:class</u>	112
B.1.10.1 allowedValueList for the <u>upnp:class</u> Property	113
B.1.10.2 <u>upnp:class@name</u>	114
B.1.11 <u>upnp:searchClass</u>	114
B.1.11.1 <u>upnp:searchClass@name</u>	114
B.1.11.2 <u>upnp:searchClass@includeDerived</u>	114
B.1.12 <u>upnp:createClass</u>	114
B.1.12.1 <u>upnp:createClass@name</u>	115
B.1.12.2 <u>upnp:createClass@includeDerived</u>	115
B.1.13 <u>upnp:writeStatus</u>	115
B.1.13.1 allowedValueList for the <u>upnp:writeStatus</u> Property	115
B.2 Resource Encoding Characteristics Properties	116
B.2.1 <u>res</u>	116
B.2.1.1 <u>res@protocolInfo</u>	116
B.2.1.2 <u>res@importUri</u>	117
B.2.1.3 <u>res@size</u>	117
B.2.1.4 <u>res@duration</u>	117
B.2.1.5 <u>res@protection</u>	117
B.2.1.6 <u>res@bitrate</u>	118
B.2.1.7 <u>res@bitsPerSample</u>	118
B.2.1.8 <u>res@sampleFrequency</u>	118
B.2.1.9 <u>res@nrAudioChannels</u>	118
B.2.1.10 <u>res@resolution</u>	118
B.2.1.11 <u>res@colorDepth</u>	118
B.2.1.12 <u>res@tspec</u>	119
B.2.1.13 <u>res@allowedUse</u>	119
B.2.1.14 <u>res@validityStart</u>	119
B.2.1.15 <u>res@validityEnd</u>	119
B.2.1.16 <u>res@remainingTime</u>	120
B.2.1.17 <u>res@usageInfo</u>	120
B.2.1.18 <u>res@rightsInfoURI</u>	120
B.2.1.19 <u>res@contentInfoURI</u>	120
B.2.1.20 <u>res@recordQuality</u>	120

B.3	Contributor-related Properties	121
B.3.1	<u>upnp:artist</u>	121
B.3.1.1	<u>upnp:artist@role</u>	121
B.3.2	<u>upnp:actor</u>	121
B.3.2.1	<u>upnp:actor@role</u>	121
B.3.3	<u>upnp:author</u>	121
B.3.3.1	<u>upnp:author@role</u>	122
B.3.4	<u>upnp:producer</u>	122
B.3.5	<u>upnp:director</u>	122
B.3.6	<u>dc:publisher</u>	122
B.3.7	<u>dc:contributor</u>	122
B.4	Affiliation-related Properties	122
B.4.1	<u>upnp:genre</u>	123
B.4.1.1	<u>upnp:genre@id</u>	123
B.4.1.2	<u>upnp:genre@extended</u>	123
B.4.2	<u>upnp:album</u>	123
B.4.3	<u>upnp:playlist</u>	123
B.5	Associated Resources Properties	123
B.5.1	<u>upnp:albumArtURI</u>	124
B.5.2	<u>upnp:artistDiscographyURI</u>	124
B.5.3	<u>upnp:lyricsURI</u>	124
B.5.4	<u>dc:relation</u>	124
B.6	Storage-Related Properties	124
B.6.1	<u>upnp:storageTotal</u>	124
B.6.2	<u>upnp:storageUsed</u>	124
B.6.3	<u>upnp:storageFree</u>	125
B.6.4	<u>upnp:storageMaxPartition</u>	125
B.6.5	<u>upnp:storageMedium</u>	125
B.6.5.1	allowedValueList for the <u>upnp:storageMedium</u> Property	125
B.7	General Description (mainly for UI purposes) Properties	125
B.7.1	<u>dc:description</u>	126
B.7.2	<u>upnp:longDescription</u>	126
B.7.3	<u>upnp:icon</u>	126
B.7.4	<u>upnp:region</u>	126
B.7.5	<u>upnp:rights</u>	126
B.7.6	<u>dc:date</u>	126
B.7.7	<u>dc:language</u>	127
B.7.8	<u>upnp:playbackCount</u>	127
B.7.9	<u>upnp:lastPlaybackTime</u>	127
B.7.10	<u>upnp:lastPlaybackPosition</u>	127
B.7.11	<u>upnp:recordedStartDateTime</u>	127
B.7.12	<u>upnp:recordedDuration</u>	127
B.7.13	<u>upnp:recordedDayOfWeek</u>	128
B.7.13.1	allowedValueList for the <u>upnp:recordedDayOfWeek</u> Property	128
B.7.14	<u>upnp:srsRecordScheduleID</u>	128
B.7.15	<u>upnp:srsRecordTaskID</u>	128
B.7.16	<u>upnp:recordable</u>	128

B.8	Recorded Object-related Properties.....	129
B.8.1	<u>upnp:programTitle</u>	129
B.8.2	<u>upnp:seriesTitle</u>	129
B.8.3	<u>upnp:programID</u>	130
B.8.3.1	<u>upnp:programID@type</u>	130
B.8.4	<u>upnp:seriesID</u>	130
B.8.4.1	<u>upnp:seriesID@type</u>	130
B.8.5	<u>upnp:channelID</u>	131
B.8.5.1	<u>upnp:channelID@type</u>	131
B.8.6	<u>upnp:episodeCount</u>	131
B.8.7	<u>upnp:episodeNumber</u>	131
B.8.8	<u>upnp:programCode</u>	132
B.8.8.1	<u>upnp:programCode@type</u>	132
B.8.9	<u>upnp:rating</u>	132
B.8.9.1	<u>upnp:rating@type</u>	132
B.8.10	<u>upnp:episodeType</u>	132
B.9	User Channel and EPG Related Properties.....	133
B.9.1	<u>upnp:channelGroupName</u>	133
B.9.1.1	<u>upnp:channelGroupName@id</u>	133
B.9.2	<u>upnp:callSign</u>	133
B.9.3	<u>upnp:networkAffiliation</u>	134
B.9.4	<u>upnp:serviceProvider</u>	134
B.9.5	<u>upnp:price</u>	134
B.9.5.1	<u>upnp:price@currency</u>	134
B.9.6	<u>upnp:payPerView</u>	134
B.9.7	<u>upnp:epgProviderName</u>	134
B.9.8	<u>upnp:dateTimeRange</u>	134
B.10	Radio Broadcast Properties.....	135
B.10.1	<u>upnp:radioCallSign</u>	135
B.10.2	<u>upnp:radioStationID</u>	135
B.10.3	<u>upnp:radioBand</u>	135
B.10.3.1	allowedValueList for the <u>upnp:radioBand</u> Property.....	135
B.11	Video Broadcast Properties.....	135
B.11.1	<u>upnp:channelNr</u>	136
B.11.2	<u>upnp:channelName</u>	136
B.11.3	<u>upnp:scheduledStartTime</u>	136
B.11.4	<u>upnp:scheduledEndTime</u>	136
B.12	Physical Tuner Status-related Properties.....	136
B.12.1	<u>upnp:signalStrength</u>	137
B.12.2	<u>upnp:signalLocked</u>	137
B.12.3	<u>upnp:tuned</u>	137
B.13	Bookmark-related Properties.....	137
B.13.1	<u>@neverPlayable</u>	138
B.13.2	<u>upnp:bookmarkID</u>	138
B.13.3	<u>upnp:bookmarkedObjectID</u>	138
B.13.4	<u>upnp:deviceUDN</u>	138
B.13.4.1	<u>upnp:deviceUDN@serviceType</u>	138
B.13.4.2	<u>upnp:deviceUDN@serviceId</u>	138
B.13.5	<u>upnp:stateVariableCollection</u>	139

B.13.5.1	<u>upnp:stateVariableCollection@serviceName</u>	139
B.13.5.2	<u>upnp:stateVariableCollection@rcsInstanceType</u>	139
B.14	Miscellaneous Properties	140
B.14.1	<u>upnp:DVDRegionCode</u>	140
B.14.2	<u>upnp:originalTrackNumber</u>	140
B.14.3	<u>upnp:toc</u>	140
B.14.4	<u>upnp:userAnnotation</u>	140
Annex C	(normative) AV Working Committee Class Definitions	141
C.1	Class Hierarchy	141
C.1.1	Class name syntax	142
C.1.2	Class Properties Overview	143
C.2	<u>object</u> (Base Class)	147
C.2.1	<u>item:object</u>	147
C.2.1.1	<u>imageItem:item</u>	147
C.2.1.2	<u>audioItem:item</u>	148
C.2.1.3	<u>videoItem:item</u>	149
C.2.1.4	<u>playlistItem:item</u>	151
C.2.1.5	<u>textItem:item</u>	152
C.2.1.6	<u>bookmarkItem:item</u>	153
C.2.1.7	<u>epgItem:item</u>	153
C.2.2	<u>container:object</u>	155
C.2.2.1	<u>person:container</u>	155
C.2.2.2	<u>playlistContainer:container</u>	156
C.2.2.3	<u>album:container</u>	157
C.2.2.4	<u>genre:container</u>	158
C.2.2.5	<u>channelGroup:container</u>	158
C.2.2.6	<u>epgContainer:container</u>	159
C.2.2.7	<u>storageSystem:container</u>	160
C.2.2.8	<u>storageVolume:container</u>	161
C.2.2.9	<u>storageFolder:container</u>	162
C.2.2.10	<u>bookmarkFolder:container</u>	162
Annex D	(normative) EBNF Syntax Definitions	163
D.1	Date&time Syntax	163
Annex E	(normative) CDS features	164
E.1	Requirements for the <i>EPG feature</i> , Version 1	165
E.2	Requirements for the <i>TUNER feature</i> , Version 1	166
E.3	Requirements for the <i>BOOKMARK feature</i> , Version 1	167

LIST OF TABLES

Table 1-1:	EBNF Operators	20
Table 1-2:	CSV Examples	21
Table 1-3:	Namespace Definitions	22
Table 1-4:	Schema-related Information	23
Table 1-5:	Default Namespaces for the AV Specifications.....	24
Table 2-1:	Properties in XML	29
Table 2-2:	State variables	33
Table 2-3:	Sort Modifiers	34
Table 2-4:	ContainerUpdateIDs Example	36
Table 2-5:	Event moderation	43
Table 2-6:	Actions	44
Table 2-7:	Arguments for <u>GetSearchCapabilities()</u>	45
Table 2-8:	Error Codes for <u>GetSearchCapabilities()</u>	45
Table 2-9:	Arguments for <u>GetSortCapabilities()</u>	45
Table 2-10:	Error Codes for <u>GetSortCapabilities()</u>	46
Table 2-11:	Arguments for <u>GetSortExtensionCapabilities()</u>	46
Table 2-12:	Error Codes for <u>GetSortExtensionCapabilities()</u>	46
Table 2-13:	Arguments for <u>GetFeatureList()</u>	46
Table 2-14:	Error Codes for <u>GetFeatureList()</u>	47
Table 2-15:	Arguments for <u>GetSystemUpdateID()</u>	47
Table 2-16:	Error Codes for <u>GetSystemUpdateID()</u>	47
Table 2-17:	Arguments for <u>Browse()</u>	48
Table 2-18:	Error Codes for <u>Browse()</u>	49
Table 2-19:	Arguments for <u>Search()</u>	50
Table 2-20:	Error Codes for <u>Search()</u>	50
Table 2-21:	Arguments for <u>CreateObject()</u>	54
Table 2-22:	Error codes for <u>CreateObject()</u>	55
Table 2-23:	Arguments for <u>DestroyObject()</u>	56
Table 2-24:	Error Codes for <u>DestroyObject()</u>	56
Table 2-25:	Update examples.....	58
Table 2-26:	Arguments for <u>UpdateObject()</u>	58
Table 2-27:	Error Codes for <u>UpdateObject()</u>	59
Table 2-28:	Arguments for <u>MoveObject()</u>	60
Table 2-29:	Error Codes for <u>MoveObject()</u>	61
Table 2-30:	Arguments for <u>ImportResource()</u>	61
Table 2-31:	Error Codes for <u>ImportResource()</u>	62
Table 2-32:	Arguments for <u>ExportResource()</u>	62

Table 2-33:	Error Codes for ExportResource()	63
Table 2-34:	Arguments for DeleteResource()	63
Table 2-35:	Error Codes for DeleteResource()	64
Table 2-36:	Arguments for StopTransferResource()	64
Table 2-37:	Error Codes for StopTransferResource()	64
Table 2-38:	Arguments for GetTransferProgress()	65
Table 2-39:	Error Codes for GetTransferProgress()	65
Table 2-40:	Arguments for CreateReference()	66
Table 2-41:	Error Codes for CreateReference()	66
Table 2-42:	Common error codes.....	67
Table B-1:	Base Properties Overview.....	110
Table B-2:	allowedValueList for the upnp:class Property.....	113
Table B-3:	allowedValueList for the upnp:writeStatus Property.....	115
Table B-4:	Resource Encoding Characteristics Properties Overview.....	116
Table B-5:	Contributor-related Properties Overview.....	121
Table B-6:	Affiliation-related Properties Overview.....	122
Table B-7:	Associated Resources Properties Overview.....	123
Table B-8:	Storage-Related Properties Overview.....	124
Table B-9:	General Description (mainly for UI purposes) Properties Overview.....	125
Table B-10:	allowedValueList for the upnp:recordedDayOfWeek Property.....	128
Table B-11:	Recorded Object-related Properties Overview.....	129
Table B-12:	User Channel and EPG Related Properties Overview.....	133
Table B-13:	Radio Broadcast Properties Overview.....	135
Table B-14:	allowedValueList for the upnp:radioBand Property.....	135
Table B-15:	Video Broadcast Properties Overview.....	135
Table B-16:	Physical Tuner Status-related Properties Overview.....	136
Table B-17:	Bookmark-related Properties Overview.....	137
Table B-18:	allowedValueList for the upnp:stateVariableCollection@rcsInstanceType Property.....	139
Table B-19:	Miscellaneous Properties Overview.....	140
Table C-1:	Class Properties Overview.....	143
Table C-2:	object Properties.....	147
Table C-3:	item Properties.....	147
Table C-4:	imageItem:item Properties.....	147
Table C-5:	photo:imageItem Properties.....	148
Table C-6:	audioItem:item Properties.....	148
Table C-7:	musicTrack:audioItem Properties.....	148
Table C-8:	audioBroadcast:audioItem Properties.....	149
Table C-9:	audioBook:audioItem Properties.....	149

Table C-10:	<u>videoItem:item</u> Properties.....	150
Table C-11:	<u>movie:videoItem</u> Properties.....	150
Table C-12:	<u>videoBroadcast:videoItem</u> Properties.....	151
Table C-13:	<u>musicVideoClip:videoItem</u> Properties.....	151
Table C-14:	<u>playlistItem:item</u> Properties	152
Table C-15:	<u>textItem:item</u> Properties	152
Table C-16:	<u>bookmarkItem:item</u> Properties	153
Table C-17:	<u>epgItem:item</u> Properties	153
Table C-18:	<u>audioProgram:epgItem</u> Properties	155
Table C-19:	<u>videoProgram:epgItem</u> Properties.....	155
Table C-20:	<u>container</u> Properties	155
Table C-21:	<u>person:container</u> Properties	155
Table C-22:	<u>musicArtist:person</u> Properties	156
Table C-23:	<u>playlistContainer:container</u> Properties.....	156
Table C-24:	<u>album:container</u> Properties	157
Table C-25:	<u>musicAlbum:album</u> Properties.....	157
Table C-26:	<u>photoAlbum:album</u> Properties	157
Table C-27:	<u>genre:container</u> Properties	158
Table C-28:	<u>channelGroup:container</u> Properties.....	159
Table C-29:	<u>epgContainer:container</u> Properties.....	160
Table C-30:	<u>storageSystem:container</u> Properties	161
Table C-31:	<u>storageVolume:container</u> Properties.....	161
Table C-32:	<u>storageFolder:container</u> Properties.....	162
Table C-33:	genre:container Properties	162
Table E-1:	<i>CDS features</i>	164
Table E-2:	REQUIRED characteristics of the <i>EPG Feature</i> element	165
Table E-3:	REQUIRED characteristics of the <i>TUNER Feature</i> element.....	166
Table E-4:	REQUIRED characteristics of the <i>BOOKMARK feature</i> element	167

LIST OF FIGURES

Figure 1: Class hierarchy for the item base class.	141
Figure 2: Class hierarchy for the container base class.	142

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008

INFORMATION TECHNOLOGY – UPNP DEVICE ARCHITECTURE –

Part 4-11: Audio Video Device Control Protocol – Level 2 – Content Directory Service

FOREWORD

- 1) 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.
- 2) 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.
- 3) 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.
- 4) 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.
- 5) 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.
- 6) 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.
- 7) All users should ensure that they have the latest edition of this publication.
- 8) 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.
- 9) 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.

IEC and ISO draw attention to the fact that it is claimed that compliance with this document may involve the use of patents as indicated below.

ISO and IEC take no position concerning the evidence, validity and scope of the putative patent rights. The holders of the putative patent rights have assured IEC and ISO that they are willing to negotiate free licences or licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of the putative patent rights are registered with IEC and ISO.

Intel Corporation has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Intel Corporation
Standards Licensing Department
5200 NE Elam Young Parkway
MS: JFS-98
USA – Hillsboro, Oregon 97124

Microsoft Corporation has informed IEC and ISO that it has patent applications or granted patents as listed below:

6101499 / US; 6687755 / US; 6910068 / US; 7130895 / US; 6725281 / US; 7089307 / US; 7069312 / US;
10/783 524 /US

Information may be obtained from:

Microsoft Corporation
One Microsoft Way
USA – Redmond WA 98052

Philips International B.V. has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Philips International B.V. – IP&S
High Tech campus, building 44 3A21
NL – 5656 Eindhoven

NXP B.V. (NL) has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

NXP B.V. (NL)
High Tech campus 60
NL – 5656 AG Eindhoven

Matsushita Electric Industrial Co. Ltd. has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Matsushita Electric Industrial Co. Ltd.
1-3-7 Shiomi, Chuoh-ku
JP – Osaka 540-6139

Hewlett Packard Company has informed IEC and ISO that it has patent applications or granted patents as listed below:

5 956 487 / US; 6 170 007 / US; 6 139 177 / US; 6 529 936 / US; 6 470 339 / US; 6 571 388 / US; 6 205 466 / US

Information may be obtained from:

Hewlett Packard Company
1501 Page Mill Road
USA – Palo Alto, CA 94304

Samsung Electronics Co. Ltd. has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Digital Media Business, Samsung Electronics Co. Ltd.
416 Maetan-3 Dong, Yeongtang-Gu,
KR – Suwon City 443-742

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. IEC and ISO shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 29341-4-12 was prepared by UPnP Implementers Corporation and adopted, under the PAS procedure, by joint technical committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

The list of all currently available parts of the ISO/IEC 29341 series, under the general title *Universal plug and play (UPnP) architecture*, can be found on the IEC web site.

This International Standard has been approved by vote of the member bodies, and the voting results may be obtained from the address given on the second title page.

ORIGINAL UPnP DOCUMENTS (informative)

Reference may be made in this document to original UPnP documents. These references are retained in order to maintain consistency between the specifications as published by ISO/IEC and by UPnP Implementers Corporation. The following table indicates the original UPnP document titles and the corresponding part of ISO/IEC 29341:

UPnP Document Title	ISO/IEC 29341 Part
UPnP Device Architecture 1.0	ISO/IEC 29341-1
UPnP Basic:1 Device	ISO/IEC 29341-2
UPnP AV Architecture:1	ISO/IEC 29341-3-1
UPnP MediaRenderer:1 Device	ISO/IEC 29341-3-2
UPnP MediaServer:1 Device	ISO/IEC 29341-3-3
UPnP AVTransport:1 Service	ISO/IEC 29341-3-10
UPnP ConnectionManager:1 Service	ISO/IEC 29341-3-11
UPnP ContentDirectory:1 Service	ISO/IEC 29341-3-12
UPnP RenderingControl:1 Service	ISO/IEC 29341-3-13
UPnP MediaRenderer:2 Device	ISO/IEC 29341-4-2
UPnP MediaServer:2 Device	ISO/IEC 29341-4-3
UPnP AV Datastructure Template:1	ISO/IEC 29341-4-4
UPnP AVTransport:2 Service	ISO/IEC 29341-4-10
UPnP ConnectionManager:2 Service	ISO/IEC 29341-4-11
UPnP ContentDirectory:2 Service	ISO/IEC 29341-4-12
UPnP RenderingControl:2 Service	ISO/IEC 29341-4-13
UPnP ScheduledRecording:1	ISO/IEC 29341-4-14
UPnP DigitalSecurityCamera:1 Device	ISO/IEC 29341-5-1
UPnP DigitalSecurityCameraMotionImage:1 Service	ISO/IEC 29341-5-10
UPnP DigitalSecurityCameraSettings:1 Service	ISO/IEC 29341-5-11
UPnP DigitalSecurityCameraStillImage:1 Service	ISO/IEC 29341-5-12
UPnP HVAC_System:1 Device	ISO/IEC 29341-6-1
UPnP HVAC_ZoneThermostat:1 Device	ISO/IEC 29341-6-2
UPnP ControlValve:1 Service	ISO/IEC 29341-6-10
UPnP HVAC_FanOperatingMode:1 Service	ISO/IEC 29341-6-11
UPnP FanSpeed:1 Service	ISO/IEC 29341-6-12
UPnP HouseStatus:1 Service	ISO/IEC 29341-6-13
UPnP HVAC_SetpointSchedule:1 Service	ISO/IEC 29341-6-14
UPnP TemperatureSensor:1 Service	ISO/IEC 29341-6-15
UPnP TemperatureSetpoint:1 Service	ISO/IEC 29341-6-16
UPnP HVAC_UserOperatingMode:1 Service	ISO/IEC 29341-6-17
UPnP BinaryLight:1 Device	ISO/IEC 29341-7-1
UPnP DimmableLight:1 Device	ISO/IEC 29341-7-2
UPnP Dimming:1 Service	ISO/IEC 29341-7-10
UPnP SwitchPower:1 Service	ISO/IEC 29341-7-11
UPnP InternetGatewayDevice:1 Device	ISO/IEC 29341-8-1
UPnP LANDevice:1 Device	ISO/IEC 29341-8-2
UPnP WANDevice:1 Device	ISO/IEC 29341-8-3
UPnP WANConnectionDevice:1 Device	ISO/IEC 29341-8-4
UPnP WLANAccessPointDevice:1 Device	ISO/IEC 29341-8-5
UPnP LANHostConfigManagement:1 Service	ISO/IEC 29341-8-10
UPnP Layer3Forwarding:1 Service	ISO/IEC 29341-8-11
UPnP LinkAuthentication:1 Service	ISO/IEC 29341-8-12
UPnP RadiusClient:1 Service	ISO/IEC 29341-8-13
UPnP WANCableLinkConfig:1 Service	ISO/IEC 29341-8-14
UPnP WANCommonInterfaceConfig:1 Service	ISO/IEC 29341-8-15
UPnP WANDSLLinkConfig:1 Service	ISO/IEC 29341-8-16
UPnP WANEthernetLinkConfig:1 Service	ISO/IEC 29341-8-17
UPnP WANIPConnection:1 Service	ISO/IEC 29341-8-18
UPnP WANPOTSLinkConfig:1 Service	ISO/IEC 29341-8-19
UPnP WANPPPoEConnection:1 Service	ISO/IEC 29341-8-20
UPnP WLANConfiguration:1 Service	ISO/IEC 29341-8-21
UPnP Printer:1 Device	ISO/IEC 29341-9-1
UPnP Scanner:1.0 Device	ISO/IEC 29341-9-2
UPnP ExternalActivity:1 Service	ISO/IEC 29341-9-10
UPnP Feeder:1.0 Service	ISO/IEC 29341-9-11
UPnP PrintBasic:1 Service	ISO/IEC 29341-9-12
UPnP Scan:1 Service	ISO/IEC 29341-9-13
UPnP QoS Architecture:1.0	ISO/IEC 29341-10-1
UPnP QoSDevice:1 Service	ISO/IEC 29341-10-10
UPnP QoSManager:1 Service	ISO/IEC 29341-10-11
UPnP QoSPolicyHolder:1 Service	ISO/IEC 29341-10-12
UPnP QoS Architecture:2	ISO/IEC 29341-11-1
UPnP QOS v2 Schema Files	ISO/IEC 29341-11-2
UPnP QoSDevice:2 Service	ISO/IEC 29341-11-10

UPnP Document Title	ISO/IEC 29341 Part
UPnP QosManager:2 Service	ISO/IEC 29341-11-11
UPnP QosPolicyHolder:2 Service	ISO/IEC 29341-11-12
UPnP RemoteUIClientDevice:1 Device	ISO/IEC 29341-12-1
UPnP RemoteUIServerDevice:1 Device	ISO/IEC 29341-12-2
UPnP RemoteUIClient:1 Service	ISO/IEC 29341-12-10
UPnP RemoteUIServer:1 Service	ISO/IEC 29341-12-11
UPnP DeviceSecurity:1 Service	ISO/IEC 29341-13-10
UPnP SecurityConsole:1 Service	ISO/IEC 29341-13-11

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008

1 Overview and Scope

This service template is compliant with the UPnP Device Architecture version 1.0. It defines a service type referred to herein as ContentDirectory service.

1.1 Introduction

Many devices within the home network contain various types of content that other devices would like to access (for example, music, videos, still images, etc). As an example, a MediaServer device might contain a significant portion of the homeowner's audio, video, and still-image library. In order for the homeowner to enjoy this content, the homeowner must be able to browse the objects stored on the MediaServer, select a specific one, and cause it to be played on an appropriate rendering device (for example, an audio player for music objects, a TV for video content, an Electronic Picture Frame for still-images, etc).

For maximum convenience, it is highly desirable to allow the homeowner to initiate these operations from a variety of UI devices. In most cases, these UI devices will either be a UI built into the rendering device, or it will be a stand-alone UI device such as a wireless PDA or tablet. In any case, it is unlikely that the homeowner will interact directly with the device containing the content (that is: the homeowner won't have to walk over to the server device). In order to enable this capability, the server device needs to provide a uniform mechanism for UI devices to browse the content on the server and to obtain detailed information about individual content objects. This is the purpose of the ContentDirectory service.

The ContentDirectory service additionally provides a lookup/storage service that allows clients (for example, UI devices) to locate (and possibly store) individual objects (for example, songs, movies, pictures, etc) that the (server) device is capable of providing. For example, this service can be used to enumerate a list of songs stored on an MP3 player, a list of still-images comprising various slide-shows, a list of movies stored in a DVD-Jukebox, a list of TV shows currently being broadcast (a.k.a an EPG), a list of songs stored in a CD-Jukebox, a list of programs stored on a PVR (Personal Video Recorder) device, etc. Nearly any type of content can be enumerated via this ContentDirectory service. For devices that contain multiple types of content (for example, MP3, MPEG2, JPEG, etc.), a single instance of the ContentDirectory service can be used to enumerate all objects, regardless of their type.

1.2 Notation

- In this document, features are described as Required, Recommended, or Optional as follows:

The key words "MUST," "MUST NOT," "REQUIRED," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," "RECOMMENDED," "MAY," and "OPTIONAL" in this specification are to be interpreted as described in [RFC 2119].

In addition, the following keywords are used in this specification:

PROHIBITED – The definition or behavior is an absolute prohibition of this specification. Opposite of **REQUIRED**.

CONDITIONALLY REQUIRED – The definition or behavior depends on a condition. If the specified condition is met, then the definition or behavior is **REQUIRED**, otherwise it is **PROHIBITED**.

CONDITIONALLY OPTIONAL – The definition or behavior depends on a condition. If the specified condition is met, then the definition or behavior is **OPTIONAL**, otherwise it is **PROHIBITED**.

These keywords are thus capitalized when used to unambiguously specify requirements over protocol and application features and behavior that affect the interoperability and security of implementations. When these words are not capitalized, they are meant in their natural-language sense.

- Strings that are to be taken literally are enclosed in "double quotes".
- Words that are emphasized are printed in *italic*.
- Keywords that are defined by the UPnP AV Working Committee are printed using the *forum* character style.
- Keywords that are defined by the UPnP Device Architecture are printed using the *arch* character style.
- A double colon delimiter, "::", signifies a hierarchical parent-child (parent::child) relationship between the two objects separated by the double colon. This delimiter is used in multiple contexts, for example: Service::Action(), Action()::Argument, parentProperty::childProperty.

1.2.1 Data Types

This specification uses data type definitions from two different sources. The UPnP Device Architecture defined data types are used to define state variable and action argument data types [DEVICE]. The XML Schema namespace is used to define property data types [XML SCHEMA-2].

For UPnP Device Architecture defined Boolean data types, it is strongly RECOMMENDED to use the value “0” for false, and the value “1” for true. However, when used as input arguments, the values “false”, “no”, “true”, “yes” may also be encountered and MUST be accepted. Nevertheless, it is strongly RECOMMENDED that all state variables and output arguments be represented as “0” and “1”.

For XML Schema defined Boolean data types, it is strongly RECOMMENDED to use the value “0” for false, and the value “1” for true. However, when used as input properties, the values “false”, “true” may also be encountered and MUST be accepted. Nevertheless, it is strongly RECOMMENDED that all properties be represented as “0” and “1”.

1.2.2 Strings Embedded in Other Strings

Some string variables and arguments described in this document contain substrings that MUST be independently identifiable and extractable for other processing. This requires the definition of appropriate substring delimiters and an escaping mechanism so that these delimiters can also appear as ordinary characters in the string and/or its independent substrings. This document uses embedded strings in two contexts – Comma Separated Value (CSV) lists (see Section 1.3.1, “Comma Separated Value (CSV) Lists”) and property values in search criteria strings. Escaping conventions use the backslash character, “\” (character code U+005C), as follows:

- a. Backslash (“\”) is represented as “\\” in both contexts.
- b. Comma (“,”) is
 1. represented as “\,” in individual substring entries in CSV lists
 2. not escaped in search strings
- c. Double quote (“””) is
 1. not escaped in CSV lists
 2. not escaped in search strings when it appears as the start or end delimiter of a property value
 3. represented as “\\” in search strings when it appears as a character that is part of the property value

1.2.3 Extended Backus-Naur Form

Extended Backus-Naur Form is used in this document for a formal syntax description of certain constructs. The usage here is according to the reference [EBNF].

1.2.3.1 Typographic conventions for EBNF

Non-terminal symbols are unquoted sequences of characters from the set of English upper and lower case letters, the digits “0” through “9”, and the hyphen (“-”). Character sequences between ‘single quotes’ are terminal strings and MUST appear literally in valid strings. Character sequences between (*comment delimiters*) are English language definitions or supplementary explanations of their associated symbols. White space in the EBNF is used to separate elements of the EBNF, not to represent white space in valid strings. White space usage in valid strings is described explicitly in the EBNF. Finally, the EBNF uses the following operators:

Table 1-1: EBNF Operators

Operator	Semantics
<code>:</code>	definition – the non-terminal symbol on the left is defined by one or more alternative sequences of terminals and/or non-terminals to its right.
<code> </code>	alternative separator – separates sequences on the right that are independently allowed definitions for the non-terminal on the left.
<code>*</code>	null repetition – means the expression to its left MAY occur zero or more times.
<code>+</code>	non-null repetition – means the expression to its left MUST occur at least once and MAY occur more times.
<code>[]</code>	optional – the expression between the brackets is optional.
<code>()</code>	grouping – groups the expressions between the parentheses.
<code>-</code>	character range – represents all characters between the left and right character operands inclusively.

1.3 Derived Data Types

This section defines a derived data type that is represented as a string data type with special syntax. This specification uses string data type definitions that originate from two different sources. The UPnP Device Architecture defined **string** data type is used to define state variable and action argument **string** data types. The XML Schema namespace is used to define property xsd:string data types. The following definition applies to both string data types.

1.3.1 Comma Separated Value (CSV) Lists

The UPnP AV services use state variables, action arguments and properties that represent lists – or one-dimensional arrays – of values. The UPnP Device Architecture, Version 1.0 [DEVICE], does not provide for either an array type or a list type, so a list type is defined here. Lists MAY either be homogeneous (all values are the same type) or heterogeneous (values of different types are allowed). Lists MAY also consist of repeated occurrences of homogeneous or heterogeneous subsequences, all of which have the same syntax and semantics (same number of values, same value types and in the same order). The data type of a homogeneous list is **string** or xsd:string and denoted by CSV (*x*), where *x* is the type of the individual values. The data type of a heterogeneous list is also **string** or xsd:string and denoted by CSV (*x*, *y*, *z*), where *x*, *y* and *z* are the types of the individual values. If the number of values in the heterogeneous list is too large to show each type individually, that variable type is represented as CSV (*heterogeneous*), and the variable description includes additional information as to the expected sequence of values appearing in the list and their corresponding types. The data type of a repeated subsequence list is **string** or xsd:string and denoted by CSV ({*x*, *y*, *z*}), where *x*, *y* and *z* are the types of the individual values in the subsequence and the subsequence MAY be repeated zero or more times.

- A list is represented as a **string** type (for state variables and action arguments) or xsd:string type (for properties).
- Commas separate values within a list.
- Integer values are represented in CSVs with the same syntax as the integer data type specified in [DEVICE] (that is: optional leading sign, optional leading zeroes, numeric ASCII)
- Boolean values are represented in state variable and action argument CSVs as either “**0**” for false or “**1**” for true. These values are a subset of the defined Boolean data type values specified in [DEVICE]: **0**, **false**, **no**, **1**, **true**, **yes**.
- Boolean values are represented in property CSVs as either “**0**” for false or “**1**” for true. These values are a subset of the defined Boolean data type values specified in [XML SCHEMA-2]: 0, false, 1, true.
- Escaping conventions for the comma and backslash characters are defined in Section 1.2.2, “Strings Embedded in Other Strings”.
- White space before, after, or interior to any numeric data type is not allowed.
- White space before, after, or interior to any other data type is part of the value.

Table 1-2: CSV Examples

Type refinement of string	Value	Comments
CSV (string) or CSV (xsd:string)	"-artist,-date"	List of 2 property sort criteria.
CSV (int) or CSV (xsd:integer)	"1,-5,006,0,+7"	List of 5 integers.
CSV (boolean) or CSV (xsd:Boolean)	"0,1,1,0"	List of 4 booleans
CSV (string) or CSV (xsd:string)	"Smith\, Fred,Jones\, Davey"	List of 2 names, "Smith, Fred" and "Jones, Davey"
CSV (i4 , string , ui2) or CSV (xsd:int, xsd:string, xsd:unsignedShort)	"-29837, string with leading blanks,0"	Note that the second value is " string with leading blanks"
CSV (i4) or CSV (xsd:int)	"3, 4"	Illegal CSV. White space is not allowed as part of an integer value.
CSV (string) or CSV (xsd:string)	" , "	List of 3 empty string values
CSV (heterogeneous)	"Alice,Marketing,5,Sue,R&D,21,Dave,Finance,7"	List of unspecified number of people and associated attributes. Each person is described by 3 elements: a name string , a department string and years-of-service ui2 or a name xsd:string, a department xsd:string and years-of-service xsd:unsignedShort.

1.4 Management of XML Namespaces in Standardized DCPs

UPnP specifications make extensive use of XML namespaces. This allows separate DCPs, and even separate components of an individual DCP, to be designed independently and still avoid name collisions when they share XML documents. Every name in an XML document belongs to exactly one namespace. In documents, XML names appear in one of two forms: qualified or unqualified. An unqualified name (or no-colon-name) contains no colon (":") characters. An unqualified name belongs to the document's default namespace. A qualified name is two no-colon-names separated by one colon character. The no-colon-name before the colon is the qualified name's namespace prefix, the no-colon-name after the colon is the qualified name's "local" name (meaning local to the namespace identified by the namespace prefix). Similarly, the unqualified name is a local name in the default namespace.

The formal name of a namespace is a URI. The namespace prefix used in an XML document is *not* the name of the namespace. The namespace name is, or should be, globally unique. It has a single definition that is accessible to anyone who uses the namespace. It has the same meaning anywhere that it is used, both inside and outside XML documents. The namespace prefix, however, in formal XML usage, is defined only in an XML document. It must be locally unique to the document. Any valid XML no-colon-name may be used. And, in formal XML usage, no two XML documents are ever required to use the same namespace prefix to refer to the same namespace. The creation and use of the namespace prefix was standardized by the W3C XML Committee in [XML-NMSP] strictly as a convenient local shorthand replacement for the full URI name of a namespace in individual documents.

All AV object properties are represented in XML by element and attribute names, therefore, all property names belong to an XML namespace.

For the same reason that namespace prefixes are convenient in XML documents, it is convenient in specification text to refer to namespaces using a namespace prefix. Therefore, this specification declares a “standard” prefix for all XML namespaces used herein. In addition, this specification expands the scope where these prefixes have meaning, beyond a single XML document, to all of its text, XML examples, and certain string-valued properties. This expansion of scope *does not* supersede XML rules for usage in documents, it only augments and complements them in important contexts that are out-of-scope for the XML specifications.

All of the namespaces used in this specification are listed in the Tables “Namespace Definitions” and “Schema-related Information”. For each such namespace, Table 1-3, “Namespace Definitions” gives a brief description of it, its name (a URI) and its defined “standard” prefix name. Some namespaces included in these tables are not directly used or referenced in this document. They are included for completeness to accommodate those situations where this specification is used in conjunction with other UPnP specifications to construct a complete system of devices and services. The individual specifications in such collections all use the same standard prefix. The standard prefixes are also used in Table 1-4, “Schema-related Information”, to cross-reference additional namespace information. This second table includes each namespace’s valid XML document root elements (if any), its schema file name, versioning information (to be discussed in more detail below), and links to the entries in the Reference section for its associated schema.

The normative definitions for these namespaces are the documents referenced in Table 1-3. The schemas are designed to support these definitions for both human understanding and as test tools. However, limitations of the XML Schema language itself make it difficult for the UPnP-defined schemas to accurately represent all details of the namespace definitions. As a result, the schemas will validate many XML documents that are not valid according to the specifications.

The Working Committee expects to continue refining these schemas after specification release to reduce the number of documents that are validated by the schemas while violating the specifications, but the schemas will still be informative, supporting documents. Some schemas might become normative in future versions of the specifications.

Table 1-3: Namespace Definitions

Standard Name-space Prefix	Namespace Name	Namespace Description	Normative Definition Document Reference
<i>AV Working Committee defined namespaces</i>			
av:	urn:schemas-upnp-org:av:av	Common data types for use in AV schemas	[AV-XSD]
avs:	urn:schemas-upnp-org:av:avs	Common structures for use in AV schemas	[AVS-XSD]
avdt:	urn:schemas-upnp-org:av:avdt	Datastructure Template	[AVDT]
avt-event:	urn:schemas-upnp-org:metadata-1-0/AVT/	Evented <i>LastChange</i> state variable for AVTransport	[AVT]
didl-lite:	urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/	Structure and metadata for ContentDirectory	[CDS]
rcs-event:	urn:schemas-upnp-org:metadata-1-0/RCS/	Evented <i>LastChange</i> state variable for RenderingControl	[RCS]
srs:	urn:schemas-upnp-org:av:srs	Metadata and structure for ScheduledRecording	[SRS]
srs-event:	urn:schemas-upnp-org:av:srs-event	Evented <i>LastChange</i> state variable for ScheduledRecording	[SRS]
upnp:	urn:schemas-upnp-org:metadata-1-0/upnp/	Metadata for ContentDirectory	[CDS]
<i>Externally defined namespaces</i>			
dc:	http://purl.org/dc/elements/1.1/	Dublin Core	[DC-TERMS]
xsd:	http://www.w3.org/2001/XMLSchema	XML Schema Language 1.0	[XML SCHEMA-1] [XML SCHEMA-2]
xsi:	http://www.w3.org/2001/XMLSchema-instance	XML Schema Instance Document schema	Sections 2.6 & 3.2.7 of [XML SCHEMA-1]
xml:	http://www.w3.org/XML/1998/namespace	The “xml:” Namespace	[XML-NS]

Table 1-4: Schema-related Information

Standard Name-space Prefix	Relative URI and File Name		Schema Reference
	• Form 1	• Form 2	
Valid Root Element(s)			
AV Working Committee Defined Namespaces			
av:	- av-vn-yyyymmdd.xsd - av-vn.xsd	n/a	[AV-XSD]
avs:	- avs-vn-yyyymmdd.xsd - avs-vn.xsd	<Features> <stateVariableValuePairs>	[AVS-XSD]
avdt:	- avdt-vn-yyyymmdd.xsd - avdt-vn.xsd	<AVDT>	[AVDT]
avt-event:	- avt-event-vn-yyyymmdd.xsd - avt-event-vn.xsd	<Event>	[AVT-EVENT-XSD]
didl-lite:	- didl-lite-vn-yyyymmdd.xsd - didl-lite-vn.xsd	<DIDL-Lite>	[DIDL-LITE-XSD]
rcs-event:	- rcs-event-vn-yyyymmdd.xsd - rcs-event-vn.xsd	<Event>	[RCS-EVENT-XSD]
srs:	- srs-vn-yyyymmdd.xsd - srs-vn.xsd	<srs>	[SRS-XSD]
srs-event:	- srs-event-vn-yyyymmdd.xsd - srs-event-vn.xsd	<StateEvent>	[SRS-EVENT-XSD]
upnp:	- upnp-vn-yyyymmdd.xsd - upnp-vn.xsd	n/a	[UPNP-XSD]
Externally Defined Namespaces			
dc:	Absolute URL: http://dublincore.org/schemas/xmls/simpledc20021212.xsd		[DC-XSD]
xsd:	n/a	<Schema>	[XMLSCHEMA-XSD]
xsi:	n/a		n/a
xml:	n/a		[XML-XSD]

1.4.1 Namespace Prefix Requirements

There are many occurrences in this specification of string data types that contain XML names (property names). These XML names in strings will not be processed under namespace-aware conditions. Therefore, all occurrences in instance documents of XML names in strings MUST use the standard namespace prefixes as declared in Table 1-3. In order to properly process the XML documents described herein, control points and devices MUST use namespace-aware XML processors [XML-NMSP] for both reading and writing. As allowed by [XML-NMSP], the namespace prefixes used in an instance document are at the sole discretion of the document creator. Therefore, the declared prefix for a namespace in a document MAY be different from the standard prefix. All devices MUST be able to correctly process any valid XML instance document, even when it uses a non-standard prefix for ordinary XML names. It is strongly RECOMMENDED that all devices use these standard prefixes for all instance documents to avoid confusion on the part of both human and machine readers. These standard prefixes are used in all descriptive text and all XML examples in this and related UPnP specifications. Also, each individual specification may assume a default namespace for its descriptive text. In that case, names from that namespace may appear with no prefix.

The assumed default namespace, if any, for each UPnP AV specification is given in Table 1-5, “Default Namespaces for the AV Specifications”.

Note: all UPnP AV schemas declare attributes to be “unqualified”, so namespace prefixes are never used with AV Working Committee defined attribute names.

Table 1-5: Default Namespaces for the AV Specifications

AV Specification Name	Default Namespace Prefix
AVTransport:2	avt-event:
ConnectionManager:2	<i>n/a</i>
ContentDirectory:2	didl-lite:
MediaRenderer:2	<i>n/a</i>
MediaServer:2	<i>n/a</i>
RenderingControl:2	rsc-event:
ScheduledRecording:1	srs:

1.4.2 Namespace Names, Namespace Versioning and Schema Versioning

Each namespace that is defined by the AV Working Committee is named by a URN.

In order to enable both forward and backward compatibility, the UPnP TC has established the general policy that namespace names will not change with new versions of specifications, even when the specification changes the definition of a namespace. But, namespaces still have version numbers that reflect definitional changes. Each time the definition of a namespace is changed, the namespace's version number is incremented by one. Therefore, namespace version information must be provided with each XML instance document so that the document's receiver can properly understand its meaning. This is achieved by the following rules:

- Every release of a schema is identified by a version number and date of the form “*n-yyyymmdd*”, where *n* corresponds to the namespace definition version number and *yyyymmdd* is the year, month and day in the Gregorian calendar that the schema is released.

For example, the new version numbers of the pre-existing “DIDL-Lite” and “upnp” schemas are “2”. Versions for new schemas, such as “srs” are “1”.

For each schema, the version-date will appear in two places:

1. In the schema file name, according to the naming structure shown in Table 1-4, “Schema-related Information”.
2. As the value of the `version` attribute of each schema's schema root element.

Namespaces are referenced in both schema and XML instance documents by namespace name. The namespace name appears as the value of an `xmlns` attribute. The `xmlns` attribute also declares a namespace prefix that will be used to qualify names from each namespace. Schemas are referenced in both schema and XML instance documents by URI in the `schemaLocation` attribute. See section 1.4.3, “Namespace Usage Examples”. Two different forms of URI are available, each with a different meaning. All UPnP AV-defined schema URIs share a common base path of “<http://www.upnp.org/schemas/av/>”. Each schema URI has two unique relative forms (see Table 1-4, “Schema-related Information”), according to which version of a namespace and its representative schema is of interest. The allowed relative URI forms are:

1. *schema-root-name* “-v” *version-date*
where *version-date* is a full version-date of the form *n-yyyymmdd*. This form references the schema whose “root” name (typically the standardized prefix name used for the namespace that the schema represents) and version-date match *schema-root-name* and *version-date*, respectively.
2. *schema-root-name* “-v” *version*
where *version* is an integer representing the namespace's version number. This form references the most recent version of the schema whose root name and namespace version number match *schema-root-name* and the *version*, respectively.

Usage rules for schema location URIs are as follows:

- All instance documents, whether generated by a service or a control point, MUST use Form 1.
- All UPnP AV published schemas that reference other UPnP AV schemas will also use Form 1.
- Validation of XML instance documents in UPnP AV systems potentially serves two purposes. The first is based on standard XML and XML Schema semantics: the document's creator asserts that the document is syntactically correct with respect to the referenced schema. The receiving processor can

confirm this with a validating parser that uses the referenced schema(s). The second is based on UPnP AV namespace semantics. The receiving processor knows that the XML instance document is supposed to conform to one or more specific UPnP AV specifications. Since the second context is actually the more important context for instance document processing, the receiving processor MAY validate the instance document against any version of a schema that satisfies its needs in assessing the acceptability of the received instance document.

1.4.3 Namespace Usage Examples

The `schemaLocation` attribute for XML instance documents comes from the XML Schema instance namespace “`http://www.w3.org/2002/XMLSchema-instance`”. A single occurrence of the attribute can declare the location of one or more schemas. The `schemaLocation` attribute value consists of a whitespace separated list of values: namespace name followed by its schema location URL. This pair-sequence is repeated as necessary for the schemas that need to be located for this instance document.

Example 1:

Sample *DIDL-Lite XML Document*. This document assumes version-date 2-20060531 of the “`didl-lite:`” namespace/schema combination and (a possible later) version 2-20061231 of “`upnp:`”. The lines with the gray background show how to express this versioning information in the instance document.

```
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20061231.xsd">
  <item id="18" parentID="13" restricted="0">
    ...
  </item>
</DIDL-Lite>
```

Example 2:

Sample *srs XML Document*. This document assumes version 1-20060531 of the “`srs:`” namespace/schema combination. Again, the lines with the gray background show how to express this versioning information in the instance document.

```
<?xml version="1.0" encoding="UTF-8"?>
<srs
  xmlns="urn:schemas-upnp-org:av:srs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:srs
    http://www.upnp.org/schemas/av/srs-v1-20060531.xsd">
  ...
</srs>
```

1.5 Vendor-defined Extensions

Whenever vendors create additional vendor-defined state variables, actions or properties, their assigned names and XML representation MUST follow the naming conventions and XML rules as specified in [DEVICE], Section 2.5, “Description: Non-standard vendor extensions”.

1.6 References

This section lists the normative references used in the UPnP AV specifications and includes the tag inside square brackets that is used for each such reference:

[AVARCH] – *AVArchitecture:1*, UPnP Forum, June 25, 2002.

Available at: <http://www.upnp.org/specs/av/UPnP-av-AVArchitecture-v1-20020625.pdf>.

[AVDT] – *AV DataStructure Template:1*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-AVDataStructure-v1-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-av-AVDataStructure-v1.pdf>.

[AVDT-XSD] – *XML Schema for UPnP AV Datastructure Template:1*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/avdt-v1-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/avdt-v1.xsd>.

[AV-XSD] – *XML Schema for UPnP AV Common XML Data Types*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/av-v1-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/av-v1.xsd>.

[AVS-XSD] – *XML Schema for UPnP AV Common XML Structures*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/avs-v1-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/avs-v1.xsd>.

[AVT] – *AVTransport:2*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-AVTransport-v2-Service-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-av-AVTransport-v2-Service.pdf>.

[AVT-EVENT-XSD] – *XML Schema for AVTransport:2 LastChange Eventing*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/avt-event-v2-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/avt-event-v2.xsd>.

[CDS] – *ContentDirectory:2*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-ContentDirectory-v2-Service-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-av-ContentDirectory-v2-Service.pdf>.

[CM] – *ConnectionManager:2*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-ConnectionManager-v2-Service-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-av-ConnectionManager-v2-Service.pdf>.

[DC-XSD] – *XML Schema for UPnP AV Dublin Core*.

Available at: <http://www.dublincore.org/schemas/xmls/simpledc20020312.xsd>.

[DC-TERMS] – *DCMI term declarations represented in XML schema language*.

Available at: <http://www.dublincore.org/schemas/xmls>.

[DEVICE] – *UPnP Device Architecture, version 1.0*, UPnP Forum, June 13, 2000.

Available at: <http://www.upnp.org/specs/architecture/UPnP-DeviceArchitecture-v1.0-20000613.htm>.

Latest version available at: <http://www.upnp.org/specs/architecture/UPnP-DeviceArchitecture-v1.0.htm>.

[DIDL] – ISO/IEC CD 21000-2:2001, *Information Technology - Multimedia Framework - Part 2: Digital Item Declaration*, July 2001.

[DIDL-LITE-XSD] – *XML Schema for ContentDirectory:2 Structure and Metadata (DIDL-Lite)*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/didl-lite-v2.xsd>.

[EBNF] – ISO/IEC 14977, *Information technology - Syntactic metalanguage - Extended BNF*, December 1996.

[HTTP/1.1] – *HyperText Transport Protocol – HTTP/1.1*, R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, June 1999.

Available at: <http://www.ietf.org/rfc/rfc2616.txt>.

[IEC 61883] – *IEC 61883 Consumer Audio/Video Equipment – Digital Interface - Part 1 to 5*.

Available at: <http://www.iec.ch>.

[IEC-PAS 61883] – *IEC-PAS 61883 Consumer Audio/Video Equipment – Digital Interface - Part 6*.

Available at: <http://www.iec.ch>.

[ISO 8601] – *Data elements and interchange formats – Information interchange -- Representation of dates and times*, International Standards Organization, December 21, 2000.

Available at: [ISO 8601:2000](http://www.iso.org/iso/8601).

[MIME] – *IETF RFC 1341, MIME (Multipurpose Internet Mail Extensions)*, N. Borenstein, N. Freed, June 1992.

Available at: <http://www.ietf.org/rfc/rfc1341.txt>.

[MR] – *MediaRenderer:2*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-MediaRenderer-v2-Device-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-AV-MediaRenderer-v2-Device.pdf>.

[MS] – *MediaServer:2*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-MediaServer-v2-Device-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-AV-MediaServer-v2-Device.pdf>.

[RCS] – *RenderingControl:2*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-RenderingControl-v2-Service-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-av-RenderingControl-v2-Service.pdf>.

[RCS-EVENT-XSD] – *XML Schema for RenderingControl:2 LastChange Eventing*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/rcs-event-v1-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/rcs-event-v1.xsd>.

[RFC 1738] – *IETF RFC 1738, Uniform Resource Locators (URL)*, Tim Berners-Lee, et. Al., December 1994.

Available at: <http://www.ietf.org/rfc/rfc1738.txt>.

[RFC 2119] – *IETF RFC 2119, Key words for use in RFCs to Indicate Requirement Levels*, S. Bradner, 1997.

Available at: <http://www.faqs.org/rfcs/rfc2119.html>.

[RFC 2396] – *IETF RFC 2396, Uniform Resource Identifiers (URI): Generic Syntax*, Tim Berners-Lee, et al, 1998.

Available at: <http://www.ietf.org/rfc/rfc2396.txt>.

[RFC 3339] – *IETF RFC 3339, Date and Time on the Internet: Timestamps*, G. Klyne, Clearswift Corporation, C. Newman, Sun Microsystems, July 2002.

Available at: <http://www.ietf.org/rfc/rfc3339.txt>.

[RTP] – *IETF RFC 1889, Realtime Transport Protocol (RTP)*, H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, January 1996.

Available at: <http://www.ietf.org/rfc/rfc1889.txt>.

[RTSP] – *IETF RFC 2326, Real Time Streaming Protocol (RTSP)*, H. Schulzrinne, A. Rao, R. Lanphier, April 1998.

Available at: <http://www.ietf.org/rfc/rfc2326.txt>.

[SRS] – *ScheduledRecording:1*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/specs/av/UPnP-av-ScheduledRecording-v1-Service-20060531.pdf>.

Latest version available at: <http://www.upnp.org/specs/av/UPnP-av-ScheduledRecording-v1-Service-20060531.pdf>.

[SRS-XSD] – *XML Schema for ScheduledRecording:1 Metadata and Structure*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/srs-v1-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/srs-v1.xsd>.

[SRS-EVENT-XSD] – *XML Schema for ScheduledRecording:1 LastChange Eventing*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/srs-event-v1-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/srs-event-v1.xsd>.

[UAX 15] – *Unicode Standard Annex #15, Unicode Normalization Forms, version 4.1.0, revision 25*, M. Davis, M. Dürst, March 25, 2005.

Available at: <http://www.unicode.org/reports/tr15/tr15-25.html>.

[UNICODE COLLATION] – *Unicode Technical Standard #10, Unicode Collation Algorithm version 4.1.0*, M. Davis, K. Whistler, May 5, 2005.

Available at: <http://www.unicode.org/reports/tr10/tr10-14.html>.

[UPNP-XSD] – *XML Schema for ContentDirectory:2 Metadata*, UPnP Forum, May 31, 2006.

Available at: <http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd>.

Latest version available at: <http://www.upnp.org/schemas/av/upnp-v2.xsd>.

[UTS 10] – *Unicode Technical Standard #10, Unicode Collation Algorithm, version 4.1.0, revision 14*, M. Davis, K. Whistler, May 5, 2005.

Available at: <http://www.unicode.org/reports/tr10/tr10-14.html>.

[UTS 35] – *Unicode Technical Standard #35, Locale Data Markup Language, version 1.3RI, revision 5*, M. Davis, June 2, 2005.

Available at: <http://www.unicode.org/reports/tr35/tr35-5.html>.

[XML] – *Extensible Markup Language (XML) 1.0 (Third Edition)*, François Yergeau, Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, eds., W3C Recommendation, February 4, 2004.

Available at: <http://www.w3.org/TR/2004/REC-xml-20040204>.

[XML-NS] – *The “xml:” Namespace*, November 3, 2004.

Available at: <http://www.w3.org/XML/1998/namespace>.

[XML-XSD] – *XML Schema for the “xml:” Namespace*.

Available at: <http://www.w3.org/2001/xml.xsd>.

[XML-NMSP] – *Namespaces in XML*, Tim Bray, Dave Hollander, Andrew Layman, eds., W3C Recommendation, January 14, 1999.

Available at: <http://www.w3.org/TR/1999/REC-xml-names-19990114>.

[XML SCHEMA-1] – *XML Schema Part 1: Structures, Second Edition*, Henry S. Thompson, David Beech, Murray Maloney, Noah Mendelsohn, W3C Recommendation, 28 October 2004.

Available at: <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>.

[XML SCHEMA-2] – *XML Schema Part 2: Data Types, Second Edition*, Paul V. Biron, Ashok Malhotra, W3C Recommendation, 28 October 2004.

Available at: <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

[XMLSCHEMA-XSD] – *XML Schema for XML Schema*.

Available at: <http://www.w3.org/2001/XMLSchema.xsd>.

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008

2 Service Modeling Definitions

2.1 Service Type

The following service type identifies a service that is compliant with this template:

urn:schemas-upnp-org:service:ContentDirectory:2

ContentDirectory service is used herein to refer to this service type.

2.2 Terms

2.2.1 object

An object is any data entity that can be returned by a ContentDirectory service from a Browse() or Search() action. The ContentDirectory service defines a class system to represent the different types of objects that are managed by the ContentDirectory service. The base class, from which all other classes are derived, is named object. The class object cannot be instantiated.

2.2.2 property

A property in the ContentDirectory service represents a characteristic of an object. Properties are distinguished by their names. The ContentDirectory service defines two kinds of properties – independent and dependent. Each independent property has zero or more dependent properties associated with it. Independent property names contain no “@” symbol; they may contain an XML namespace prefix (see below for an explanation of the relationship between properties and XML). Each dependent property is associated either with exactly one independent property or directly with a ScheduledRecording service class. The name of a dependent property that is associated with an independent property is the concatenation of three parts: its associated independent property name, the “@” symbol, and a name for the relationship between the two properties’ values. The name of a dependent property that is associated directly with a class is just the “@” symbol followed by the relationship name. A small number of independent properties have child independent properties. These are indicated by the concatenation of parent property name, “::”, child property name. The data types and meanings for all properties are defined in Annex B.

Even though ContentDirectory service properties are not XML objects, XML is used to express them in all exchanges between a control point and a ContentDirectory service implementation. This creates an unavoidable relationship between XML syntax and property names and values. In XML, an independent property is represented as an element. The property name is used as the element name. The property value is the element content. A child property is represented as an element in the content of the parent property element. A dependent property is represented as an attribute in XML. The dependent property’s relationship name is used as the attribute name. The dependent property’s value is the attribute value. For dependent properties that are associated with an independent property, the attribute appears in the start tag of the element that represents its associated independent property. For dependent properties that are associated directly with a class, the attribute appears in the top-level start tag for each object of that class.

Examples:

Table 2-1: Properties in XML

Property Name	XML Representation (didl-lite declared as default namespace)
<u>dc:title</u>	<dc:title>...</dc:title>
<u>res</u>	<res>...</res>
<u>res@size</u>	<res size="...">...</res>
<u>@id</u>	<item id="...">...</item>

2.2.2.1 Multi-valued property

Some independent properties are multi-valued. This means that the property MAY occur more than once in an object.

2.2.2.2 Single-valued property

Most independent properties are single-valued. This means that the property MUST occur at most once in an object. Some single-valued properties can contain a CSV list of values. A dependent property is always considered single-valued, because it can occur at most once with each occurrence of its associated independent property, even though the independent property may be multi-valued.

2.2.3 class

A class is used to assign a type to an object. It also identifies the minimum REQUIRED set of properties that MUST be present on that object and the OPTIONAL properties that MAY be present. Classes are organized in a hierarchy with certain classes being derived from others as in a typical object-oriented system. At the root of the class hierarchy is the object base class. Examples are object.item.audioItem.musicTrack and object.container.album.musicAlbum. See Section C.1.1, “Class name syntax” for a definition of the format of the class specification for an object.

2.2.4 item

item is a first-level class derived directly from object. An item most often represents a single piece of AV data, such as a CD track, a movie or an audio file. Items MAY be playable, meaning they have information that can be played on a rendering device. Any object which is derived from the item class is represented in XML using the DIDL-Lite element <item>...</item>.

Note: The term item is used in this specification to indicate an object whose class is either item or any of the defined item-derived classes.

2.2.5 container

container is a first-level class derived directly from object. A container instance represents a collection of objects. Containers can represent the physical organization of objects (storage containers) or logical collections. Logical collections can have formal definitions of their contents or they can be arbitrary collections. Containers can be either homogeneous, containing objects that are all of the same class, or heterogeneous, containing objects of mixed class. Containers can contain other containers. Any object derived from the container class is represented in XML using the DIDL-Lite element <container>...</container>.

A ContentDirectory service is REQUIRED to maintain a ContainerUpdateID for each of its containers. This value is maintained internally, does not appear in any XML expression of the container, and cannot be used in a search or sort criterion.

Note: The term container is used in this specification to indicate an object whose class is either container or any of the defined container-derived classes.

2.2.6 container modification

A container (including any object whose class is derived from the container class) is considered modified when any of the following occurs:

- A property of the container is added, removed or changed in value.
- A direct child object (including vendor defined objects) is added to or removed from the container.
- A direct child object (including vendor defined objects) has one of its properties added, removed or changed.

Note to implementers: since ContainerUpdateID is not a formal property of a container, a modification to a direct child container that affects that child's ContainerUpdateID does not propagate upward to the parent container.

2.2.7 XML Document

An XML document is a string that represents a valid XML 1.0 document according to a specific schema. Every occurrence of the phrase “*XML Document*” is italicized and preceded by the document’s root element name (also italicized), as listed in Table 1-4, “Schema-related Information”.

For example, the phrase *DIDL-Lite XML Document* refers to a valid XML 1.0 document according to the DIDL-Lite schema [DIDL-LITE-XSD]. Such a document comprises a single <DIDL-Lite ...> root element, optionally preceded by the XML declaration <?xml version="1.0" ...?>.

This string will therefore be of one of the following two forms:

“<DIDL-Lite ...>...</DIDL-Lite>”

or

“<?xml ...?><DIDL-Lite ...>...</DIDL-Lite>”

2.2.8 XML Fragment

An XML fragment is a sequence of XML elements that are valid direct or indirect child elements of the root element according to a specific schema. Every occurrence of the phrase “*XML Fragment*” is italicized and preceded by the document’s root element name (also italicized), as listed in Table 1-4, “Schema-related Information”.

The following are examples of *DIDL-Lite XML Fragments*:

“<item id="..." ...>...</item>”

or

“<res protocolInfo="..." ...>...</res>”

or

“<dc:title>Sunrise</dc:title>”

2.2.9 ContainerUpdateID

An unsigned integer associated with each Instance of class *container*. The integer value is incremented each time the container is modified (see the entry in this table for the precise definition of container *modification*). Upon reaching the value $2^{32}-1$, the next update rolls the value back to zero. The initial value of ContainerUpdateID for any newly created container is unspecified, but RECOMMENDED to be zero. Implementers SHOULD maintain the same value for each container’s ContainerUpdateID through power cycles and any other disappearance/reappearance on the network. ContainerUpdateID is not a formal property of a container object, so a modification to a direct child container that affects that child’s ContainerUpdateID does not propagate upward to the parent container.

2.2.10 *reference, reference item, referenced item*

A *reference* is a link from one ContentDirectory service item to another. It allows one item (the *reference item*) to provide a copy of the metadata of another item (the *referenced item*) without having to actually create a physical copy of the metadata. In addition to eliminating duplicate physical copies of the *referenced item*’s metadata, a reference enables a *reference item* to automatically *track* metadata changes in the *referenced items*. For example, if there are three playlist containers that all contain child items representing the same song, the ContentDirectory service implementation may store one item that contains all of the song’s metadata and store two (smaller) *reference items* that simply point to the one *referenced item*.

When a *reference item* is browsed (via Browse() or Search() actions), it MUST be returned as a valid DIDL-Lite object (for example, @id, dc:title, etc. properties are REQUIRED). The metadata of the returned object is an exact copy of the metadata from the *referenced item* with the following exceptions:

- The *reference item* MUST have a unique @id property value.
- The *reference item* MUST contain a @refID property, whose value MUST be equal to the value of the @id property of the *referenced item*. Note: Control points may use the existence of the @refID property to distinguish between a *referenced item* and all of the *reference items* that point to it.
- The *reference item* MAY have a different @parentID property value.

- The *reference item* MAY (as described below) override (for example, change a property value or remove an existing property) any of the original *referenced item*'s properties.

Additionally, a reference item MAY override any of the original referenced item's properties in one of the following ways:

- A *reference item* may be updated so that its metadata includes one or more additional properties not present in the *referenced item*.
- A *reference item* may be updated so that its metadata does not contain one or more of the existing properties of the *referenced item*.
- A *reference item* may be updated so that its metadata overrides the value of one or more existing properties of the *referenced item*.

All of the modifications listed above are bound to the *reference item* and MUST NOT propagate back to the *referenced item*, that is: the original *referenced item* MUST NOT be affected by any modifications of the *reference item*. All resulting changes specified by the *reference item* MUST result in a valid DIDL-Lite object when subsequently browsed and/or searched.

2.2.11 CDS feature

The *CDS feature* exposes extended functionality of a ContentDirectory service implementation. Each *CDS feature* has a normative name, such as "*EPG*", and a set of requirements to realize the feature. These requirements are defined in Annex E.

2.3 State Variables

Unlike most other services, the ContentDirectory service is primarily action-based. The service state variables exist primarily to support argument passing in the service actions. Information is not exposed directly through explicit state variables. Rather, a client retrieves ContentDirectory service information via the return arguments of the actions defined below. The majority of state variables defined below exist simply to enable the various actions of this service.

Reader Note: For a first-time reader, it may be more helpful to read the action definitions before reading the state variable definitions.

2.3.1 State Variable Overview

Table 2-2: State variables

Variable Name	R/O ¹	Data Type	Allowed Value	Default Value	Eng. Units
<u>SearchCapabilities</u>	<u>R</u>	<u>string</u>	CSV (<u>string</u>) See Section 2.3.2		
<u>SortCapabilities</u>	<u>R</u>	<u>string</u>	CSV (<u>string</u>) See Section 2.3.3		
<u>SortExtensionCapabilities</u>	<u>O</u> ²	<u>string</u>	CSV (<u>string</u>) See Section 2.3.4		
<u>SystemUpdateID</u>	<u>R</u>	<u>ui4</u>	See Section 0		
<u>ContainerUpdateIDs</u>	<u>O</u>	<u>string</u>	CSV ({ <u>string</u> , <u>ui4</u> }) See Section 2.3.6		
<u>TransferIDs</u>	<u>O</u>	<u>string</u>	CSV (<u>ui4</u>) See Section 2.3.7		
<u>FeatureList</u>	<u>R</u>	<u>string</u>	<i>Features XML Document</i> See Section 2.3.8		
<u>A_ARG_TYPE_ObjectID</u>	<u>R</u>	<u>string</u>	See Section 2.3.9		
<u>A_ARG_TYPE_Result</u>	<u>R</u>	<u>string</u>	See Section 2.3.10		
<u>A_ARG_TYPE_SearchCriteria</u>	<u>O</u>	<u>string</u>	See Section 2.3.11		
<u>A_ARG_TYPE_BrowseFlag</u>	<u>R</u>	<u>string</u>	BrowseMetadata, BrowseDirectChildren		
<u>A_ARG_TYPE_Filter</u>	<u>R</u>	<u>string</u>	CSV (<u>string</u>) See Section 2.3.13		
<u>A_ARG_TYPE_SortCriteria</u>	<u>R</u>	<u>string</u>	CSV (<u>string</u>) See Section 2.3.14		
<u>A_ARG_TYPE_Index</u>	<u>R</u>	<u>ui4</u>	See Section 2.3.15		
<u>A_ARG_TYPE_Count</u>	<u>R</u>	<u>ui4</u>	See Section 2.3.16		
<u>A_ARG_TYPE_UpdateID</u>	<u>R</u>	<u>ui4</u>	See Section 2.3.17		
<u>A_ARG_Type_TransferID</u>	<u>O</u>	<u>ui4</u>	See Section 2.3.18		
<u>A_ARG_Type_TransferStatus</u>	<u>O</u>	<u>string</u>	See Section 2.3.19		
<u>ARG_Type_TransferLength</u>	<u>O</u>	<u>string</u>	See Section 2.3.20		
<u>A_ARG_Type_TransferTotal</u>	<u>O</u>	<u>string</u>	See Section 2.3.21		
<u>A_ARG_TYPE_TagValueList</u>	<u>O</u>	<u>string</u>	CSV (<u>string</u>) See Section 2.3.22		
<u>A_ARG_TYPE_URI</u>	<u>O</u>	<u>uri</u>	See Section 2.3.23		
<i>Non-standard state variables implemented by a UPnP vendor go here</i>	<u>X</u>	<i>TBD</i>	<i>TBD</i>	<i>TBD</i>	<i>TBD</i>

¹ R = REQUIRED, O = OPTIONAL, X = Non-standard

² REQUIRED if implementation supports sort modifiers other than “+” and “-”

2.3.2 SearchCapabilities

This state variable is a CSV list of property names that can be used in search queries. An empty string indicates that the ContentDirectory service does not support any kind of searching. A wildcard (“*”) indicates that the device supports search queries using all property names supported by the ContentDirectory service.

2.3.3 SortCapabilities

This state variable is a CSV list of property names that the ContentDirectory service can use to sort Search() or Browse() action results. An empty string indicates that the device does not support any kind of sorting. A wildcard (“*”) indicates that the device supports sorting using all property names supported by the ContentDirectory service.

2.3.4 SortExtensionCapabilities

This state variable is a CSV list of sort modifiers that the ContentDirectory service can use to sort Search() or Browse() results. Table 2-3, “Sort Modifiers” defines the standard sort modifiers. Other standard sort modifiers MAY be defined in future versions of this specification. Vendors MAY define vendor-specific sort modifiers.

Modifiers MUST be treated as case-sensitive.

Omitting this state variable is identical to listing only “+” and “-” modifiers. If modifiers other than “+” and “-” are supported, this state variable is REQUIRED.

Table 2-3: Sort Modifiers

Sort Modifiers	Descriptions
<u>+</u> , <u>-</u>	The “+” and “-” modifiers indicate that the sort is in ascending or descending order, respectively, with regard to the value of its associated property. The modifiers “+” and “-” MUST be supported by any service that supports sorting. Sorting support is indicated by a non-empty value for the <u>SortCapabilities</u> state variable. When a ContentDirectory service implements the <u>SortExtensionCapabilities</u> state variable, the values “+” and “-” MUST be included.
<u>TIME+</u> , <u>TIME-</u>	The “<u>TIME+</u>” and “<u>TIME-</u>” modifiers indicate the sort is in ascending or descending order, respectively, with regard to only the time part value of the date format property. For example, sorting on “<u>TIME+</u>dc:date” results in the following response. Either both of these modifiers MUST be supported or neither of them. If a time zone offset is included in the property’s value, it MUST be accounted for in the sort results. If no time zone offset is included, the time value is assumed to be local time. - Object A that has “2004-05-08T10:00:00” in <u>dc:date</u>. - Object C that has “2004-05-11T12:00:00” in <u>dc:date</u>. - Object B that has “2003-02-12T18:30:00” in <u>dc:date</u>. - Object D that has “2003-02-10” in <u>dc:date</u>. As shown above, some objects may not have a value for the time part in the specified property. In that case, such objects MUST appear before the other sorted results in ascending order or after in descending order. If no time value is present the implementation MUST assume that nothing is known about it. This is <i>not</i> equivalent to a time value of “00:00:00”.
<i>Vendor defined</i>	Vendors MAY add sort modifiers.

2.3.5 SystemUpdateID

This REQUIRED state variable changes whenever anything in the ContentDirectory service changes. A change could be a new or removed object, or a change in the metadata of an object. This variable is evented and the event is moderated at a maximum rate of 5 Hz (once every 0.2 seconds). The actual value of SystemUpdateID is unspecified. However, implementers SHOULD maintain the same value for SystemUpdateID through power cycles and any other disappearance/reappearance of the service on the network. Control points can use a change in the value of this variable to determine if there has been a change in the ContentDirectory service.

Note that the (OPTIONAL) ContainerUpdateIDs variable provides more information about the scope of the change, since it takes advantage of the ContainerUpdateID values maintained for each container.

2.3.6 ContainerUpdateIDs

This OPTIONAL state variable is an unordered CSV list of ordered pairs. Each pair consists of a ContainerID and a ContainerUpdateID, in that order, separated by a comma (“,”). ContainerUpdateIDs is a moderated evented state variable and is *only* used for eventing. There is no action that returns the value of ContainerUpdateIDs. The initial value of ContainerUpdateIDs is the empty string.

Each time a container is modified (see container modification in 2.2.6, “container modification”), its ContainerUpdateID is incremented and the ordered pair of ContainerID and ContainerUpdateID values is concatenated to the list ContainerUpdateIDs. If the ContainerID already appears in ContainerUpdateIDs, the new ordered pair is *not* added to the list. Instead, the corresponding ContainerUpdateID that is already in ContainerUpdateIDs is replaced by the new ContainerUpdateID value. Consequently, there can be at most one occurrence in ContainerUpdateIDs of an ordered pair with any specific ContainerID. ContainerUpdateIDs is not a history list of container changes. Its evented value will never show the same ContainerID, ContainerUpdateID value pair twice, and a subscribing control point only sees the last value of ContainerUpdateID immediately prior to the event.

The net effect for a ContentDirectory service implementation is that the ContainerUpdateIDs state variable is not cleared immediately after it has been evented. Rather, the ContainerUpdateIDs state variable is cleared immediately *before* the first new ContainerID, ContainerUpdateID pair is added to the ContainerUpdateIDs state variable following a ContainerUpdateIDs event. The reason for this behavior is that if the ContainerUpdateIDs state variable were to be cleared immediately after eventing, then when the current moderation period ends, the empty list would be evented (because the ContainerUpdateIDs state variable changed since the last event). This would falsely indicate a state change in the ContentDirectory service that did not actually occur.

Example: The following table shows a time-ordered sequence of actions on a ContentDirectory service for container modifications.

Table 2-4: ContainerUpdateIDs Example

Action	<u>ContainerID</u>	New value of <u>ContainerUpdateID</u>	
		↓	New value of <u>ContainerUpdateIDs</u>
Initialization	—	—	"" (empty)
container modified	musicAlbum15	53	"musicAlbum15,53"
container modified	photoAlbum28	427	"musicAlbum15,53,photoAlbum28,427"
container modified	musicAlbum15	54	"musicAlbum15,54,photoAlbum28,427"
container modified	musicAlbum11	12	"musicAlbum15,54,photoAlbum28,427, musicAlbum11,12"
<u>ContainerUpdateIDs</u> is evented	—	—	Value does not change.
New control point signs up for events	—	—	Value does not change. The special event value unicast to the new control point includes the full set of 3 pairs
container modified	musicAlbum01	97	Value is first cleared, then set to "musicAlbum01,97"

If this OPTIONAL state variable is not supported, a control point may use the Browse() or Search() actions to query the ContainerUpdateID for an individual container. For indications of ContentDirectory service-wide change, the evented state variable SystemUpdateID or the action GetSystemUpdateID() can be used.

2.3.7 TransferIDs

This state variable is a CSV list of type A_ARG_TYPE_TransferID. It is evented to notify clients when file transfers initiated by ImportResource() or ExportResource() started or finished. When a file transfer starts, its transfer ID is added to the TransferIDs list. When the transfer ends, its ID is removed from the TransferIDs list.

This state variable is used for eventing only.

2.3.8 FeatureList

This state variable enumerates the *CDS features* supported by this ContentDirectory service. The value is a valid *Features XML Document*, according to [AVS-XSD]:

- The root element of the document is <Features>. It contains zero or more child Feature elements, each of which represents one ContentDirectory service feature that is supported in this implementation.
- A <Features> element MUST have a version attribute.
- A <Feature> element MAY have other attributes defined per each feature.
- See the schema in [AVS-XSD] for more details on the structure.

Example (this string must be escaped when transmitted in a SOAP response message):

```
<?xml version="1.0" encoding="UTF-8"?>
<Features
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <Feature name="BOOKMARK" version="1">
    <objectIDs>bookmark1</objectIDs>
  </Feature>
  <Feature name="EPG" version="1">
    <objectIDs>epg1,epg2</objectIDs>
  </Feature>
</Features>
```

2.3.9 **A ARG TYPE ObjectID**

This state variable is introduced to provide type information for the *ObjectID* argument in various actions. The *ObjectID* argument uniquely identifies individual objects within the ContentDirectory service.

2.3.10 **A ARG TYPE Result**

This state variable is introduced to provide type information for the *Result* argument in various actions. The structure of the *Result* argument is a *DIDL-Lite XML Document*:

- Optional XML declaration `<?xml version="1.0" ?>`
- `<DIDL-Lite>` is the root element.
- `<container>` is the element representing objects of class *container* and all its derived classes.
- `<item>` is the element representing objects of class *item* and all its derived classes.
- Elements in the Dublin Core (dc) and UPnP (upnp) namespaces represent object metadata.
- See the DIDL-Lite schema [DIDL-LITE-XSD] for more details on the structure. The available properties and their names are described in Annex B, “**(normative)**

AV Working Committee Properties”.

Note that since the value of *Result* is XML, it needs to be escaped (using the normal XML rules: [XML] Section 2.4 Character Data and Markup) before embedding in a SOAP response message. In addition, when a value of type *A ARG TYPE Result* is employed in a CSV list, commas (“,”) that appear within XML CDATA MUST be escaped as “\,”. See Section 1.2.2, “Strings Embedded in Other Strings”

2.3.11 **A ARG TYPE SearchCriteria**

This state variable is introduced to provide type information for the *SearchCriteria* argument in the *Search()* action. The *SearchCriteria* argument provides one or more search criteria to be used for querying the ContentDirectory service.

2.3.11.1 **SearchCriteria String Syntax**

SearchCriteria string syntax is described here formally using EBNF as described in Section 1.2.3, “Extended Backus-Naur Form”.

```

searchCrit ::= searchExp | asterisk
searchExp ::= relExp |
             searchExp wChar+ logOp wChar+ searchExp |
             '(' wChar* searchExp wChar* ')'
logOp      ::= 'and' | 'or'
relExp     ::= property wChar+ binOp wChar+ quotedVal |
             property wChar+ existsOp wChar+ boolVal
binOp      ::= relOp | stringOp
relOp      ::= '=' | '!=' | '<' | '<=' | '>' | '>='
stringOp   ::= 'contains' | 'doesNotContain' | 'derivedfrom'
existsOp   ::= 'exists'
boolVal    ::= 'true' | 'false'
quotedVal  ::= dQuote escapedQuote dQuote
wChar     ::= space | hTab | lineFeed | vTab | formFeed | return
property   ::= (* property name as defined in Section 2.4 *)
escapedQuote ::= (* double-quote escaped string as defined in Section
1.2.2 *)

```

```

hTab      ::= (* UTF-8 code 0x09, horizontal tab character *)
lineFeed  ::= (* UTF-8 code 0x0A, line feed character *)
vTab      ::= (* UTF-8 code 0x0B, vertical tab character *)
formFeed  ::= (* UTF-8 code 0x0C, form feed character *)
return    ::= (* UTF-8 code 0x0D, carriage return character *)
space     ::= ' '
           (* UTF-8 code 0x20, space character *)
dQuote    ::= '"'
           (* UTF-8 code 0x22, double quote character *)
asterisk   ::= '*'
           (* UTF-8 code 0x2A, asterisk character *)

```

2.3.11.2 SearchCriteria String Semantics and Examples

- **Operator precedence**

Precedence, highest to lowest, is:

```

dQuote
( )
binOp, existsOp
and
or

```

Examples:

“s1 and s2 or s3 or s4 and s5”

is equivalent to:

“(s1 and s2) or s3) or (s4 and s5)”

Likewise,

“s1 and s2 or (s3 or s4) and s5”

is equivalent to:

“(s1 and s2) or ((s3 or s4) and s5)”

- **Return all.** The special value “*” means *find everything, or return all objects that exist beneath the selected starting container.*
- **Property existence testing.** Property existence is queried for by using the `exists` operator. Strictly speaking, `exists` could be a unary operator. This SearchCriteria syntax makes it a binary operator to simplify search string parsing – there are no unary operators. The string “actor exists true” is true for every object that has at least one occurrence of the actor property. It is false for any object that has no actor property. Similarly, the string “actor exists false” is false for every object that has at least one occurrence of the actor property. It is true for any object that has no actor property.
- **Property omission.** Any property value query (as distinct from an existence query) applied to an object that does not have that property evaluates to false.
- **Class derivation testing.** Existence of objects whose class is derived from some base class specification is queried for by using the `derivedfrom` operator. For example:

“upnp:class derivedfrom "object.item"” is true for all objects whose class is object.item, or whose class name begins with object.item.
- **Numeric comparisons.** When the operator in a `relExp` is a `relOp`, and both the `escapedQuote` value and the actual property value are sequences of decimal digits or sequences of decimal digits preceded by either a “+” or “-” sign (that is: integers), the comparison is done numerically. For all other combinations of operators and property values, the comparison is done by treating both values as strings, converting a numeric value to its string representation in decimal if necessary.

Note: The ContentDirectory service is not expected to recognize any kind of numeric data other than decimal integers, composed only of decimal digits with the optional leading sign.

- **String comparisons.** All operators when applied to strings use case-insensitive comparisons.

2.3.12 **A ARG TYPE BrowseFlag**

This state variable is introduced to provide type information for the BrowseFlag argument in the Browse() action. A BrowseFlag argument specifies a browse option to be used for browsing the ContentDirectory service. Valid values are:

BrowseMetadata - this indicates that the properties of the object specified by the ObjectID argument will be returned in the Result argument.

BrowseDirectChildren - this indicates that first level objects under the object specified by the ObjectID argument will be returned in the Result argument, as well as the metadata of all objects specified.

2.3.13 **A ARG TYPE Filter**

This state variable is introduced to provide type information for the Filter argument in the Browse() and Search() actions. The comma-separated list of property specifiers (including namespaces) indicates which metadata properties are to be returned in the Result of the Browse() or Search() actions.

The Filter argument allows control points to control the complexity of the object metadata properties that are returned within the DIDL-Lite Result argument of the Browse() and Search() actions. Properties REQUIRED by the DIDL-Lite schema are always returned in the Result output argument. The Filter argument allows a control point to specify additional properties, not REQUIRED by the DIDL-Lite schema to be returned in Result. Compliant ContentDirectory service implementations do not return optional properties unless they are explicitly requested in the Filter input argument.

Both independent and dependent properties MAY be included in the comma-separated Filter argument.

If the Filter argument is equal to “*”, all supported properties, both REQUIRED and OPTIONAL, from all namespaces are returned.

A compliant ContentDirectory service implementation MUST also ignore optional properties requested in the Filter input argument, which are not actually present in the matching objects. For example, a Browse() Filter input argument of the form “dc:creator” is successful and returns a DIDL-Lite Result value that complies with the other Browse() input arguments, even in the case that the objects represented in Result do not have a dc:creator property defined.

In all cases, a compliant ContentDirectory service implementation MUST always respond to Search() and Browse() requests with the smallest, valid *DIDL-Lite XML Document* in the Result argument that satisfies the Filter input argument. In some cases, a ContentDirectory service MUST add properties that are not specified in the Filter argument so that the resulting *DIDL-Lite XML Document* is valid. If the XML document can not be made valid by adding other properties, the offending properties in the Filter argument MUST be ignored by the ContentDirectory service.

Example 1: The Filter argument in a Search() action is specified as “didl-lite:res@size”, indicating that the optional res@size property, if present, MUST be returned in the results of the Search().

Request :

```
Search("0", "dc:title contains "tenderness"", "didl-lite:res@size", 0, 1,
  "")
```

The ContentDirectory service responds with the smallest, valid *DIDL-Lite XML Document* in the Result argument that satisfies the Filter argument, as follows:

Response :

```
Search(
  <?xml version="1.0" encoding="UTF-8"?>
  <DIDL-Lite
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
    xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
```

```
urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
  http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
urn:schemas-upnp-org:metadata-1-0/upnp/
  http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
<item id="18" parentID="13" restricted="0">
  <dc:title>Try a little tenderness</dc:title>
  <upnp:class>object.item.audioItem.musicTrack</upnp:class>
  <res protocolInfo="http-get:*:audio/mpeg:*" size="3558000">
    http://168.192.1.1/audio197.mp3
  </res>
</item>
</DIDL-Lite>", 1, 1, 2345)
```

By the same token, individual properties NOT specified in the comma-separated *Filter* list that are REQUIRED for a valid DIDL-Lite *Result* are automatically included. In Example 1, since *dc:title* and *upnp:class* are REQUIRED properties for both item and container objects, the *dc:title* and *upnp:class* elements are automatically included in all item and container objects in the *Result*. The REQUIRED *res@protocolInfo* property is also automatically included in the *Result*. (Note that the REQUIRED inclusion of the dependent *res@protocolInfo* property forces the inclusion of its associated independent *res* property.)

Example 2: The *Filter* argument in a *Search()* action is specified as "upnp:longDescription,dc:creator", indicating that the optional *upnp:longDescription* and *dc:creator* properties MUST be included in the DIDL-Lite *Result* returned for each object.

Request :

```
Search("0", "dc:title contains \"tenderness\"",
      "upnp:longDescription,dc:creator", 0, 1, "")
```

The ContentDirectory service responds with the smallest, valid *DIDL-Lite XML Document* that satisfies the other *Search()* arguments and the specified *Filter* argument, as follows:

Response :

```
Search("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
      http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
      http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="18" parentID="13" restricted="0">
    <dc:title>Try a little tenderness</dc:title>
    <upnp:class>object.item.audioItem.musicTrack</upnp:class>
    <upnp:longDescription>
      This song is considered to be the finest R&B tune ever
    </upnp:longDescription>
    <dc:creator>Otis Redding</dc:creator>
  </item>
</DIDL-Lite>", 1, 1, 2345)
```

2.3.14 ***A ARG TYPE SortCriteria***

This state variable is introduced to provide type information for the *SortCriteria* argument in the *Browse()* and *Search()* actions. *A ARG TYPE SortCriteria* is a possibly empty CSV list of property names, each of which MUST be prefixed by a sort modifier. Sort modifiers indicate whether the prefixed property is to be sorted in ascending or descending order. They may also indicate that the sort process should use some special interpretation of the property's value. See Section 2.3.4 "*SortExtensionCapabilities*" for detailed information about sort modifiers. Properties appear in the list in order of descending sort priority. For example, a value of

"+upnp:artist,-dc:date,+dc:title"

would sort first by artist in ascending order, then within each artist by date in descending order (most recent first) and finally by title in ascending order.

When a device receives a SortCriteria argument using unsupported sort modifiers, it MUST return with error code 709, “Unsupported or invalid sort criteria”.

When a SortCriteria argument contains property names of optional and/or multi-valued or CSV list properties, the following rules apply:

If the property is prefixed by “+” then:

- Objects that do not have a value for the property are returned first in their group.
- Objects that have at least one value for the property are returned next in their group. Objects that have multiple values for the property (either multi-valued or CSV list) are sorted based on the property value that would cause the object to appear earliest in the list.

If the property is prefixed by “-” then:

- Objects that have at least one value for the property are returned first in their group. Objects that have multiple values (either multi-valued or CSV list) for the property are sorted based on the property value that would cause the object to appear earliest in the list.
- Objects that do not have a value for the property are returned last in their group.

Depending on the property, the sort operation uses the semantics of the property, rather than the alphabetical order of the values of that property. Note that alphabetical sorting should not be based on Unicode character values but rather based on localized lexical conventions. For example, the “ö” character in German sorts between “n” and “p” characters whereas in Swedish, it sorts after “z”. See [UNICODE COLLATION].

When an empty string is specified, then the order is device dependent. Additionally, this device dependent ordering MUST remain constant unless the SystemUpdateID value has changed since the last action invocation. In other words, any two objects that appear in a Result argument MUST always appear in the same relative order as long as the SystemUpdateID value did not change.

Note that only properties available in SortCapabilities can be sorted on.

2.3.15 A ARG TYPE Index

This state variable is introduced to provide type information for the Index argument in various actions. Index arguments specify an offset into an arbitrary list of objects. A value of 0 represents the first object in the list.

2.3.16 A ARG TYPE Count

This state variable is introduced to provide type information for the Count argument in various actions. Count arguments specify an ordinal number of arbitrary objects.

2.3.17 A ARG TYPE UpdateID

This state variable is introduced to provide type information for the UpdateID output argument in the Browse() and Search() actions. The returned value will either be the SystemUpdateID (see Section 0, “

SystemUpdateID”) or a ContainerUpdateID (see Section 2.2, “Terms”).

2.3.18 **A ARG TYPE TransferID**

This state variable is introduced to provide type information for the TransferID argument in various actions. The TransferID argument uniquely identifies individual file transfers initiated by the ImportResource() or the ExportResource() action of the ContentDirectory service. The TransferID is a unique value assigned by the device.

2.3.19 **A ARG TYPE TransferStatus**

This state variable is introduced to provide type information for the TransferStatus argument in various actions. This variable MAY assume one of the enumerated values: “IN_PROGRESS”, “STOPPED”, “ERROR”, or “COMPLETED”, indicating the status of a file transfer.

2.3.20 **A ARG TYPE TransferLength**

This state variable is introduced to provide type information for the TransferLength argument in various actions. Its data type is string, representing a numerical value that MAY exceed 32 bits in size.

2.3.21 **A ARG TYPE TransferTotal**

This state variable is introduced to provide type information for the TransferTotal argument in various actions. Its data type is string, representing a numerical value that MAY exceed 32 bits in size.

2.3.22 **A ARG TYPE TagValueList**

This state variable is introduced to provide type information for the CurrentTagValue and NewTagValue arguments in the UpdateObject() action. It is a CSV list of pairs of *DIDL-Lite XML fragments*. Each fragment is either an empty placeholder or a well-formed XML element. Note that commas (“,”) that appear within XML CDATA in the fragments MUST be escaped (as “\,”). (See Section 1.3.1, “Comma Separated Value (CSV) Lists”).

2.3.23 **A ARG TYPE URI**

This state variable is introduced to provide type information for the URI argument in various actions. URI IN or OUT arguments in ContentDirectory service actions MUST be properly escaped URIs as described in [RFC 2396]. In addition, URI arguments MUST be escaped according to the requirements of RFC1738 (<http://www.ietf.org/rfc/rfc1738.txt>).

2.4 Eventing and Moderation

Table 2-5: Event moderation

Variable Name	Evented	Moderated Event	Max Event Rate ¹	Logical Combination	Min Delta per Event ²
<u>TransferIDs</u>	<u>YES</u>	<u>NO</u>			
<u>A_ARG_TYPE_ObjectID</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_Result</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_SearchCriteria</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_SortCriteria</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_UpdateID</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_BrowseFlag</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_Filter</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_Index</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_Count</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_Type_TransferID</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_Type_TransferStatus</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_Type_TransferLength</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_Type_TransferTotal</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_TagValueList</u>	<u>NO</u>	<u>NO</u>			
<u>A_ARG_TYPE_URI</u>	<u>NO</u>	<u>NO</u>			
<u>SearchCapabilities</u>	<u>NO</u>	<u>NO</u>			
<u>SortCapabilities</u>	<u>NO</u>	<u>NO</u>			
<u>SortExtensionCapabilities</u>	<u>NO</u>	<u>NO</u>			
<u>FeatureList</u>	<u>NO</u>	<u>NO</u>			
<u>SystemUpdateID</u>	<u>YES</u>	<u>YES</u>	0.2 seconds		
<u>ContainerUpdateIDs</u>	<u>YES</u>	<u>YES</u>	0.2 seconds		
<i>Non-standard state variables implemented by a UPnP vendor go here</i>	<i>TBD</i>	<i>TBD</i>	<i>TBD</i>	<i>TBD</i>	<i>TBD</i>

¹ Determined by N, where Rate = (Event)/(N seconds).
² (N) * (allowedValueRange Step)

2.5 Actions

The following tables and subsections define the various ContentDirectory service actions.

Except where noted, if an invoked action returns an error, the state of the device will be unaffected.

Table 2-6: Actions

Name	R/O ¹
<u>GetSearchCapabilities()</u>	<u>R</u>
<u>GetSortCapabilities()</u>	<u>R</u>
<u>GetSortExtensionCapabilities()</u>	<u>Q</u> ²
<u>GetFeatureList()</u>	<u>R</u>
<u>GetSystemUpdateID()</u>	<u>R</u>
<u>Browse()</u>	<u>R</u>
<u>Search()</u>	<u>Q</u>
<u>CreateObject()</u>	<u>Q</u>
<u>DestroyObject()</u>	<u>Q</u>
<u>UpdateObject()</u>	<u>Q</u>
<u>MoveObject()</u>	<u>Q</u>
<u>ImportResource()</u>	<u>Q</u>
<u>ExportResource()</u>	<u>Q</u>
<u>DeleteResource()</u>	<u>Q</u>
<u>StopTransferResource()</u>	<u>Q</u>
<u>GetTransferProgress()</u>	<u>Q</u>
<u>CreateReference()</u>	<u>Q</u>
<i>Non-standard actions implemented by an UPnP vendor go here.</i>	<u>X</u>

¹ R = REQUIRED, Q = OPTIONAL, X = Non-standard

² REQUIRED if implementation supports sort modifiers other than “+” and “-”

Note: Non-standard actions MUST be implemented in such a way that they do not interfere with the basic operation of the ContentDirectory service, that is: these actions MUST be OPTIONAL and do not need to be invoked for the ContentDirectory service to operate normally.

2.5.1 GetSearchCapabilities()

This action returns the searching capabilities that are supported by the device.

2.5.1.1 Arguments

Table 2-7: Arguments for GetSearchCapabilities()

Argument	Direction	Related State Variable
<u>SearchCaps</u>	<u>OUT</u>	<u>SearchCapabilities</u>

2.5.1.2 Dependency on State

None.

2.5.1.3 Effect on State

None.

2.5.1.4 Errors

Table 2-8: Error Codes for GetSearchCapabilities()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.

2.5.2 GetSortCapabilities()

This action returns a CSV list of property names that can be used in the sortCriteria argument. The property names returned MUST include the appropriate namespace prefixes, except for the didl-lite namespace. Properties in the didl-lite namespace MUST always be returned without the prefix.

2.5.2.1 Arguments

Table 2-9: Arguments for GetSortCapabilities()

Argument	Direction	Related State Variable
<u>SortCaps</u>	<u>OUT</u>	<u>SortCapabilities</u>

2.5.2.2 Dependency on State

None.

2.5.2.3 Effect on State

None.

2.5.2.4 Errors

Table 2-10: Error Codes for [GetSortCapabilities\(\)](#)

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.

2.5.3 [GetSortExtensionCapabilities\(\)](#)

This action returns the CSV list of sort modifiers supported by the ContentDirectory service. This action MUST be implemented if modifiers other than “+” and “-” are supported.

2.5.3.1 Arguments

Table 2-11: Arguments for [GetSortExtensionCapabilities\(\)](#)

Argument	Direction	Related State Variable
<u>SortExtensionCaps</u>	<u>OUT</u>	<u>SortExtensionCapabilities</u>

2.5.3.2 Dependency on State

None.

2.5.3.3 Effect on State

None.

2.5.3.4 Errors

Table 2-12: Error Codes for [GetSortExtensionCapabilities\(\)](#)

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.

2.5.4 [GetFeatureList\(\)](#)

This action returns a *Features XML Document* describing which optional *CDS features* this device supports, if any.

2.5.4.1 Arguments

Table 2-13: Arguments for [GetFeatureList\(\)](#)

Argument	Direction	Related State Variable
<u>FeatureList</u>	<u>OUT</u>	<u>FeatureList</u>

2.5.4.2 Dependency on State

None.

2.5.4.3 Effect on State

None.

2.5.4.4 Errors

Table 2-14: Error Codes for GetFeatureList()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.

2.5.5 GetSystemUpdateID()

This action returns the current value of state variable SystemUpdateID. It can be used by clients that want to poll for any changes in the ContentDirectory service (as opposed to subscribing to events).

2.5.5.1 Arguments

Table 2-15: Arguments for GetSystemUpdateID()

Argument	Direction	Related State Variable
<u>Id</u>	<u>OUT</u>	<u>SystemUpdateID</u>

2.5.5.2 Dependency on State

None.

2.5.5.3 Effect on State

None.

2.5.5.4 Errors

Table 2-16: Error Codes for GetSystemUpdateID()

Error Code	Error Description	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.

2.5.6 Browse()

This action allows the caller to incrementally browse the *native* hierarchy of the ContentDirectory service objects exposed by the ContentDirectory service, including information listing the classes of objects available in any particular object container.

2.5.6.1 Arguments

The following list presents an overview of the Browse() action arguments.

- ObjectID: The @id of the object currently being browsed. An ObjectID value of zero corresponds to the root object of the ContentDirectory service.
- BrowseFlag: See Section 2.3.12, “A_ARG_TYPE BrowseFlag.”
- Filter: See Section 2.3.13, “A_ARG_TYPE Filter.”
- StartingIndex: Zero-based offset to enumerate children under the container specified by ObjectID. StartingIndex MUST be set to 0 if BrowseFlag is equal to “BrowseMetaData”.

- **RequestedCount**: Requested number of entries under the object specified by **ObjectID**. **RequestedCount** = 0 indicates request all entries.
- **SortCriteria**: See Section 2.3.14, “**A_ARG_TYPE_SortCriteria**.”
- **Result**: See Section 2.3.10, “**A_ARG_TYPE_Result**.”
- **NumberReturned**: Number of objects returned in the **Result** argument. If **BrowseFlag** is set to “**BrowseMetadata**”, then **NumberReturned** MUST be set to 1.
- **TotalMatches**: If **BrowseFlag** is set to “**BrowseMetadata**”, then **TotalMatches** MUST be set to 1. Else if **BrowseFlag** is set to “**BrowseDirectChildren**”, then **TotalMatches** MUST be set to the total number of objects in the object specified for the **Browse()** action (independent of the starting index specified by the **StartingIndex** argument).
If the ContentDirectory service implementation cannot timely compute the value of **TotalMatches**, but there are matching objects that have been found by the ContentDirectory service implementation, then the **Browse()** action MUST successfully return with the **TotalMatches** argument set to zero and the **NumberReturned** argument indicating the number of returned objects.
If the ContentDirectory service implementation cannot timely compute the value of **TotalMatches**, and there are no matching objects found, then the **Browse()** action MUST return error code 720.
- **UpdateID**: If a container’s **@id** is specified in the **ObjectID** argument, then the **ContainerUpdateID** (see Section 2.2, “Terms”) of that container is returned. If the control point has a **ContainerUpdateID** for the container that is not equal to the **UpdateID** last returned, then the control point should refresh all its state relative to that container. If the **ObjectID** argument does not refer to a container, then the value returned in the **UpdateID** argument MUST be the **SystemUpdateID**.

Table 2-17: Arguments for **Browse()**

Argument	Direction	Related State Variable
ObjectID	IN	A_ARG_TYPE_ObjectID
BrowseFlag	IN	A_ARG_TYPE_BrowseFlag
Filter	IN	A_ARG_TYPE_Filter
StartingIndex	IN	A_ARG_TYPE_Index
RequestedCount	IN	A_ARG_TYPE_Count
SortCriteria	IN	A_ARG_TYPE_SortCriteria
Result	OUT	A_ARG_TYPE_Result
NumberReturned	OUT	A_ARG_TYPE_Count
TotalMatches	OUT	A_ARG_TYPE_Count
UpdateID	OUT	A_ARG_TYPE_UpdateID

2.5.6.2 Dependency on State

None.

2.5.6.3 Effect on State

None.

2.5.6.4 Errors

Table 2-18: Error Codes for **Browse()**

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	No such object	The <u>Browse()</u> request failed because the specified <u>ObjectID</u> argument is invalid.
709	Unsupported or invalid sort criteria	The <u>Browse()</u> request failed because the specified <u>SortCriteria</u> is not supported or is invalid.
720	Cannot process the request	The <u>Browse()</u> request failed because the ContentDirectory service is unable to compute, in the time allotted, the total number of objects that are a match for the browse criteria and is additionally unable to return, in the time allotted, any objects that match the browse criteria.

2.5.7 **Search()**

This action allows the caller to search the ContentDirectory service for objects that match some search criteria. The search criteria are specified as a query string operating on properties with comparison and logical operators.

2.5.7.1 Arguments

The **Filter**, **StartingIndex**, **RequestedCount**, **SortCriteria** input arguments are the same as the corresponding input arguments for the **Browse()** action. The **Result** and **UpdateID** output arguments are the same as the corresponding output arguments for the **Browse()** action. (See Section 2.5.6, “**Browse()**”). In addition, the following arguments are defined:

- **ContainerID**: Unique identifier of the container in which to begin searching. A **ContainerID** value of zero corresponds to the root object of the ContentDirectory service.
- **NumberReturned**: Number of ContentDirectory service objects returned in the **Result** argument.
- **TotalMatches**: Total number of ContentDirectory service objects that match the search criteria (specified by the **SearchCriteria** argument, and independent of the starting index specified by the **StartingIndex** argument) under the object specified by the **ContainerID** argument.
If the ContentDirectory service implementation cannot timely compute the value of **TotalMatches**, but there are matching objects that have been found by the ContentDirectory service implementation, then the **Search()** action MUST successfully return with the **TotalMatches** argument set to zero and the **NumberReturned** argument indicating the number of returned objects.
If the ContentDirectory service implementation cannot timely compute the value of **TotalMatches**, and there are no matching objects found, then the **Search()** action MUST return error code 720.
- **SearchCriteria**: See Section 2.3.11, “**A_ARG_TYPE_SearchCriteria**.”

Table 2-19: Arguments for Search()

Argument	Direction	Related State Variable
<u>ContainerID</u>	<u>IN</u>	<u>A_ARG_TYPE_ObjectID</u>
<u>SearchCriteria</u>	<u>IN</u>	<u>A_ARG_TYPE_SearchCriteria</u>
<u>Filter</u>	<u>IN</u>	<u>A_ARG_TYPE_Filter</u>
<u>StartingIndex</u>	<u>IN</u>	<u>A_ARG_TYPE_Index</u>
<u>RequestedCount</u>	<u>IN</u>	<u>A_ARG_TYPE_Count</u>
<u>SortCriteria</u>	<u>IN</u>	<u>A_ARG_TYPE_SortCriteria</u>
<u>Result</u>	<u>OUT</u>	<u>A_ARG_TYPE_Result</u>
<u>NumberReturned</u>	<u>OUT</u>	<u>A_ARG_TYPE_Count</u>
<u>TotalMatches</u>	<u>OUT</u>	<u>A_ARG_TYPE_Count</u>
<u>UpdateID</u>	<u>OUT</u>	<u>A_ARG_TYPE_UpdateID</u>

2.5.7.2 Dependency on State

None.

2.5.7.3 Effect on State

None.

2.5.7.4 Errors

Table 2-20: Error Codes for Search()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
708	Unsupported or invalid search criteria	The <u>Search()</u> request failed because the <u>SearchCriteria</u> argument is not supported or is invalid
709	Unsupported or invalid sort criteria	The <u>Search()</u> request failed because the <u>SortCriteria</u> argument is not supported or is invalid
710	No such container	The <u>Search()</u> request failed because the <u>ContainerID</u> argument is invalid or identifies an object that is not a container.
720	Cannot process the request	The <u>Search()</u> request failed because the ContentDirectory service is unable to compute, in the time allotted, the total number of objects that are a match for the search criteria and is additionally unable to return, in the time allotted, any objects that match the search criteria.

2.5.8 CreateObject()

This action creates a new object in the container identified by ContainerID. The Elements input argument MUST conform to the DIDL-Lite schema [DIDL-LITE-XSD]. Consequently, the minimum information that MUST be included is the @id, @parentID, @restricted, dc:title, and upnp:class properties. Since the value of the @id property is assigned by the CreateObject() action, it MUST initially be set to "". The value of the @parentID property MUST match the value specified by the ContainerID input argument. Additionally, the @restricted property MUST be set to "0" (false). If any of these requirements are not met, the device MUST return error code 712 – "Bad metadata".

The ContentDirectory service MUST prevent control points from initializing the @restricted property to "1" (true) since restricted objects can only be deleted and/or modified by the service itself according to some service-internal rules. Allowing a control point to initialize the @restricted property to "1" would create an object that can not be deleted and/or modified by the service because the service does not know the rules for that object. If this were allowed to happen, the new object would become both permanent and un-modifiable.

The other properties of the new object are initialized according to the specified input properties. In addition, the ContentDirectory service MAY create additional properties, for example, to ensure consistency across the whole directory. The unique @id assigned to the newly created object is returned in the output argument ObjectID. The complete object description is returned in output argument Result in DIDL-Lite form.

2.5.8.1 res Property Creation

When the new object will have one or more res properties, the res properties MUST be generated in one of two ways:

- **The control point specifies a value of the res property and other known associated res@xxx properties.** The value of the res property MUST identify a pre-existing resource, for example, an Internet radio station. When a res value is present, the resource is available immediately and there is no need to invoke the ImportResource() action.

The following is an example of the res property and its associated res@xxx properties as returned in the Result argument of the CreateObject() action when the control point specifies a value for both the res property and the res@protocolInfo property:

Request :

```
CreateObject("10", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="" parentID="10" restricted="0">
    <dc:title>New Song</dc:title>
    <upnp:class>
      object.item.audioItem
    </upnp:class>
    <res protocolInfo="http*:audio/mp3:*">
      http://10.0.0.1/contentdir?id=10
    </res>
  </item>
</DIDL-Lite>")
```

Response :

```
CreateObject ("12", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="12" parentID="10" restricted="0">
    <dc:title>New Song</dc:title>
    <dc:creator></dc:creator>
    <upnp:class>
      object.item.audioItem
    </upnp:class>
    <res protocolInfo="http*:audio/mp3:*">
      http://10.0.0.1/contentdir?id=10
    </res>
  </item>
</DIDL-Lite>")
```

If the ContentDirectory service implementation allows the resource to be updated, then in addition, the [res@importUri](#) property will be returned. It can be used to *update* the resource at a later stage (using the [ImportResource\(\)](#) action):

```
...
<res
  protocolInfo="http*:audio/mp3:*"
  importUri="http://10.0.0.1/postdir?id=10">
    http://10.0.0.1/contentdir?id=10
</res>
...
```

- **The control point does not specify a value for the [res](#) property.** In this case, the ContentDirectory service MUST create the [res@importUri](#) property whose value is used for importing the resource at a later time. The [res](#) property returned to the control point (as part of the [CreateObject\(\)](#) [Result](#) output argument) has no value (actually set to ""). The resource is therefore not yet accessible.
To make the content accessible, one of the following must occur for each [res](#) property that does not have a value:
 - a. Some external entity (for example, the device that has an external copy of the desired content) uses the value of the associated [res@importURI](#) property to push (for example, via HTTP-POST) the desired content to the ContentDirectory service implementation. This creates a local copy of the external content.
 - b. The control point invokes the [ImportResource\(\)](#) action with the [SourceURI](#) argument set to the external location of the desired content and the [DestinationURI](#) argument set to the value of the associated [res@importUri](#) property of the target item. The [ImportResource\(\)](#) action uses HTTP-GET on the [SourceURI](#) to retrieve the target content and to create a local copy of it. The [DestinationURI](#) argument (which is set to the value of the associated [res@importURI](#) property) is simply used to uniquely identify the local destination location that will receive the content.

The following is an example of the [res](#) property and its associated [res@xxx](#) properties as returned in the [Result](#) argument of the [CreateObject\(\)](#) action when the control point does not specify a value for the [res](#) property, but provides the [res@protocolInfo](#) property and a value for the [res@importUri](#) property that MUST used to bind the resource to the object:

Request :

```

CreateObject ("10", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="" parentID="10" restricted="0">
    <dc:title>New Song</dc:title>
    <upnp:class>
      object.item.audioItem
    </upnp:class>
    <res protocolInfo="*:*:*:*">
    </res>
  </item>
</DIDL-Lite>")

```

Response :

```

CreateObject ("12", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="12" parentID="10" restricted="0">
    <dc:title>New Song</dc:title>
    <dc:creator></dc:creator>
    <upnp:class>
      object.item.audioItem
    </upnp:class>
    <res
      protocolInfo="http*:audio/mp3:*"
      importUri="http://10.0.0.1/postdir?id=10">
      http://10.0.0.1/contentdir?id=10
    </res>
  </item>
</DIDL-Lite>")

```

Once the local copy has been created, the ContentDirectory service implementation sets the value of the res property to a URI that resolves to the new local copy, and the content is then accessible. This new content URI MAY be different from the value of the associated res@importURI property. The ContentDirectory service implementation MAY subsequently remove the associated res@importURI property, or keep it for the purpose of updating the content in the future.

In both cases (1. and 2.), if the control point knows the MIME-type of the resource being added, the associated res@protocolInfo property should be set to “*:*:MIME-type:*” (for example, “*:*:audio/m3u:”). Otherwise, it should be set to “*:*:*:*”. It is the responsibility of the ContentDirectory service to fill in the appropriate values for the protocol, network and additionalInfo fields of the associated res@protocolInfo property (for example, “http*:audio/m3u:”) when it knows them (typically after importing the resource). This information is used to enable compatibility checking between MediaServer and MediaRenderer devices for this resource.

The ability to allow *res* properties in container objects is vendor dependent. If a ContentDirectory service implementation does not allow container objects to have *res* properties, attempting to create a container object with a *res* property MUST generate error code 712 – “Bad metadata”.

CreateObject() can not be used to create *reference items*. *Reference items* are actually references to other existing ContentDirectory service items and are generated with the *CreateReference()* action.

CreateObject() can also be used to create a new bookmark item. A bookmark item can be created in any container. When a bookmark item is created, the associated content item MUST be updated so that one of its *upnp:bookmarkID* properties contains the object ID of the newly created bookmark item. After the bookmark is created, it MUST contain the object ID of the bookmarked content item in its *upnp:bookmarkedObjectID* property. (See also Appendix E.3, “Requirements for the *BOOKMARK* feature, Version 1”)

In the *Elements* input argument, the *upnp:bookmarkedObjectID*, *dc:title*, *upnp:deviceUDN* (AVT and RCS), *upnp:deviceUDN@serviceId*, *upnp:deviceUDN@serviceType*, and *upnp:stateVariableCollection* properties are specified to create a bookmark item. The *upnp:class* property MUST be set to “*object.item.bookmarkItem*” or a derived class if the *upnp:createClass* property in the bookmark container allows this. Other bookmark related information such as creation time (*dc:date*) is created by the ContentDirectory service or the control point. If the control point has a clock, it sets creation time to the current time. If available, the ContentDirectory service can overwrite the control point-supplied creation time with its own notion of creation time. If the ContentDirectory service does not have a clock, then it MUST NOT update or remove creation time from the object. Table C-16, “*bookmarkItem:item* Properties” shows the structure of each bookmark item.

Note:

1. AVTransport service implementations that want to participate in scenarios that use bookmarks MUST implement the *AVTransportURIMetaData* state variable to store the relevant *DIDL-Lite XML* fragment that includes the object ID of the current content.
2. A control point embedded with a private MediaServer or MediaRenderer MUST provide a persistent UDN that is not exposed to the network but is used in a data structure that contains a UDN field. Additionally, serviceType and serviceId must be supported by the device.

Vendors who want to enhance a bookmark application can add vendor-specific fields to bookmark items.

In addition, the *CreateObject()* action can be used to create a new bookmark container. The newly created container MUST have the bookmark container class type and SHOULD set the *@neverPlayable* property to “1” if it will never contain content other than (non-playable) bookmarks.

2.5.8.2 Arguments

Table 2-21: Arguments for *CreateObject()*

Argument	Direction	Related State Variable
<i>ContainerID</i>	<i>IN</i>	<i>A_ARG_TYPE_ObjectID</i>
<i>Elements</i>	<i>IN</i>	<i>A_ARG_TYPE_Result</i>
<i>ObjectID</i>	<i>OUT</i>	<i>A_ARG_TYPE_ObjectID</i>
<i>Result</i>	<i>OUT</i>	<i>A_ARG_TYPE_Result</i>

2.5.8.3 Dependency on State

None.

2.5.8.4 Effect on State

This action updates the *SystemUpdateID* and the *ContainerUpdateID* for this container. Also, if creation of this object causes other containers to change, their *ContainerUpdateID* values are updated as well.

2.5.8.5 Errors

Table 2-22: Error codes for CreateObject()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
710	No such container	<u>CreateObject()</u> failed because the <u>ContainerID</u> argument is invalid or identifies an object that is not a container.
712	Bad metadata	<u>CreateObject()</u> failed because the <u>Elements</u> argument is not supported or is invalid.
713	Restricted parent object	<u>CreateObject()</u> failed because the <u>@restricted</u> property of the container specified by <u>ContainerID</u> argument is set to “ <u>1</u> ”.

2.5.9 DestroyObject()

This action destroys the specified object when permitted. If the object is a container, all of its child objects are also deleted, recursively. Each deleted object becomes invalid and all references to it are also deleted.

The results of DestroyObject() in the case that the targeted object is a container with @restricted property set to “0” and one or more direct child or descendant child objects with @restricted properties set to “1” are vendor-dependent. There are three likely outcomes when this condition prevails:

- The ContentDirectory service implementation destroys the specified container as well as all direct child and descendant objects of the specified container, regardless of whether or not they are restricted. The DestroyObject() action returns successfully.
- The ContentDirectory service implementation does not destroy any objects. The DestroyObject() action fails and returns error code 711.
- The ContentDirectory service implementation does not destroy the specified container, but does destroy all of the *non-restricted* direct child and descendant objects of the specified container that are not needed to preserve the original object structure hierarchy, and returns successfully.

Because the results of the DestroyObject() action are vendor dependent when the above condition prevails, control points are strongly recommended to execute DestroyObject() on all of the descendant and child objects in the targeted container object individually before attempting to destroy the container.

The ContentDirectory service implementation MAY delete a resource when it detects, with absolute certainty, that there are no references to it left anywhere in the ContentDirectory service after the successful DestroyObject() action. For ContentDirectory service implementations that *do not* attempt to delete resources, DestroyObject() returns successfully. These ContentDirectory service implementations may possess some means of handling resources that are no longer referenced by the ContentDirectory service as a result of the DestroyObject() action. For ContentDirectory service implementations that *do* attempt to delete resources, there are three likely outcomes of the DestroyObject() action:

- The ContentDirectory service implementation deletes all or some portion of the resources that are no longer referenced. The DestroyObject() action returns successfully even if only a portion or no resources at all are deleted.
- DestroyObject() fails and returns error code 714 indicating that an unsuccessful attempt was made to delete one or more resources referenced in the target object because one or more resources were not found. No resources are deleted and there is no change in state of the ContentDirectory service.
- DestroyObject() fails and returns error code 715 indicating that an unsuccessful attempt was made to delete one or more resources referenced in the target object because one or more resources can not be accessed. No resources are deleted and there is no change in state of the ContentDirectory service.

This action can also be used to destroy a bookmark item or a bookmark container. When a bookmark item is to be destroyed, the ContentDirectory service MUST first find the associated content item using the upnp:bookmarkedID property of the bookmark item and it MUST remove the associated upnp:bookmarkID property from the content item. Similarly, when a content item that contains one or more upnp:bookmarkID properties is destroyed, the ContentDirectory service MUST find all associated bookmark items and MUST also delete those bookmark items.

2.5.9.1 Arguments

Table 2-23: Arguments for DestroyObject()

Argument	Direction	Related State Variable
<u>ObjectID</u>	<u>IN</u>	<u>A_ARG_TYPE_ObjectID</u>

2.5.9.2 Dependency on State

None.

2.5.9.3 Effect on State

This action updates the SystemUpdateID and the ContainerUpdateID of the parent container. Also, if deletion of this object causes other containers to change, and those containers exist after the completion of the DestroyObject() action, their ContainerUpdateID values are updated as well.

2.5.9.4 Errors

Table 2-24: Error Codes for DestroyObject()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	No such object	<u>DestroyObject()</u> failed because the object specified by the <u>ObjectID</u> argument is invalid.
711	Restricted object	<u>DestroyObject()</u> failed because the <u>@restricted</u> property of the object specified by the <u>ObjectID</u> argument is set to “ <u>I</u> ”.
713	Restricted parent object	<u>DestroyObject()</u> failed because the <u>@restricted</u> property of the parent object of the object specified by the <u>ObjectID</u> argument is set to “ <u>I</u> ”.
714	No such resource	<u>DestroyObject()</u> failed because the resource referenced by the object specified by the <u>ObjectID</u> argument cannot be identified.
715	Source resource access denied	<u>DestroyObject()</u> failed because the resource referenced by the object specified by the <u>ObjectID</u> argument cannot be accessed.

2.5.10 UpdateObject()

This action modifies, deletes or inserts object metadata. The object to be updated is specified by ObjectID. The CurrentTagValue argument is a CSV list of *DIDL-Lite XML fragments*. Each fragment is either the complete, exact, current text of the XML representation of an existing independent property of the object or an empty placeholder. The NewTagValue argument is also a CSV list of *DIDL-Lite XML fragments*, each of which is the complete new XML representation of an independent property for the object or an empty placeholder. The two tag/value lists MUST have the same number of entries. Each entry in the CurrentTagValue argument represents properties to be modified. The corresponding entry in the NewTagValue represents the new, replacement value for the property identified by the CurrentTagValue argument.

Identification of existing properties to be modified MUST be exact. The CurrentTagValue entry text MUST exactly match the text of the property value. This is easily done by copying the text from a previously returned Search() or Browse() result, and it allows UpdateObject() to perform a simple string comparison between a CurrentTagValue entry and the current information. The matched existing property is then replaced by the corresponding value in the NewTagValue argument. The independent property name in the NewTagValue entry MUST be the same as in the CurrentTagValue entry. Using UpdateObject() to replace one property by a different property is not allowed. This MUST be accomplished by first deleting the old property and then adding the new one.

If an entry in the CurrentTagValue argument is an empty string, it matches the *null* element, and the net effect is to insert the corresponding NewTagValue entry as a new property. Conversely, if an entry in the CurrentTagValue argument is not empty but the corresponding NewTagValue entry is, the net effect is to remove the current property. If changing or removing a property would result in an object that no longer satisfies the requirements as specified by the object's class, the UpdateObject() action MUST fail without any change and return the appropriate error code. Examples include:

- Removing a REQUIRED property, unless the property appears multiple times and this single removal leaves the object with a valid set of REQUIRED occurrences.
- Changing the value of the dc:date property to a person's name.
- Changing the object's class.

The ContentDirectory service MAY delete a resource when it detects, with absolute certainty, that there are no remaining references to the resource anywhere in the ContentDirectory service after the UpdateObject() action.

When the specified object is targeted to acquire one or more res properties as part of the update, the res properties MUST be generated as specified in Section 2.5.8.1, "res Property Creation".

When the two argument lists contain two or more entries, the multiple-update request is performed as an atomic operation. In other words, all modifications to the object MUST be made before any change is visible to an external observer. Either the entire request succeeds or no property is changed. A partial update MUST never occur.

Items that are actually references to other, existing ContentDirectory service items are generated with the CreateReference() action. *Reference items* cannot be generated using the UpdateObject() action.

Objects can be moved within the ContentDirectory service hierarchy with the MoveObject() action. Objects cannot be moved with UpdateObject() because the @parentID property is a dependent property and UpdateObject() does not provide a way to change a dependent property without sending the entire value of its associated independent property. In this case, it would be the entire object.

Table 2-25: Update examples

Operation	<u>CurrentTagValue</u>	<u>NewTagValue</u>	Notes
Change the title of a song	<dc:title> My Favorite Song </dc:title>	<dc:title> My Second Favorite Song </dc:title>	
Delete the date property	<dc:date> 1990-01-01 </dc:date>	(Empty String)	
Insert a genre tag	(Empty String)	<upnp:genre> Swing </upnp:genre>	
Change the artist's name from Singer1 to Singer2	<upnp:artist> Singer1 </upnp:artist>	<upnp:artist > Singer2 </upnp:artist >	The entire top-level tag (that is: <upnp:artist>) is included in both <u>CurrentTagValue</u> and <u>NewTagValue</u> .
Change the title, insert another genre, and delete the publisher property	<dc:title> My Favorite Song </dc:title>,, <dc:publisher> Acme Music </dc:publisher>	<dc:title> My Third Favorite Song </dc:title>, <upnp:genre>Jazz </upnp:genre>,,	In <u>CurrentTagValue</u> , note the double-comma placeholder just after </dc:title>. In <u>NewTagValue</u> , note that the trailing comma at the end is a placeholder for the deleted <dc:publisher> tag.

2.5.10.1 Arguments**Table 2-26: Arguments for UpdateObject()**

Argument	Direction	Related State Variable
<u>ObjectID</u>	<u>IN</u>	<u>A_ARG_TYPE_ObjectID</u>
<u>CurrentTagValue</u>	<u>IN</u>	<u>A_ARG_TYPE_TagValueList</u>
<u>NewTagValue</u>	<u>IN</u>	<u>A_ARG_TYPE_TagValueList</u>

2.5.10.2 Dependency on State

None.

2.5.10.3 Effect on State

This action will change the metadata of the specified object. It also updates the SystemUpdateID state variable and the ContainerUpdateID value for the parent container. Finally, if changing this object causes other containers to change, and those containers exist after the completion of the action, their ContainerUpdateID values are updated as well.

2.5.10.4 Errors

Table 2-27: Error Codes for UpdateObject()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	No such object	<u>UpdateObject()</u> failed because the specified <u>ObjectID</u> is invalid.
702	Invalid currentTagValue	<u>UpdateObject()</u> failed because the tag/value pair(s) listed in <u>CurrentTagValue</u> do not match the current state of the ContentDirectory service. The specified data is likely out of date.
703	Invalid newTagValue	<u>UpdateObject()</u> failed because the specified value for the <u>NewTagValue</u> argument is invalid.
704	Required tag	<u>UpdateObject()</u> failed because the request included a request to delete a required tag.
705	Read only tag	<u>UpdateObject()</u> failed because the request included a request to update a read only tag.
706	Parameter Mismatch	<u>UpdateObject()</u> failed because the number of tag/value pairs (including empty placeholders) in <u>CurrentTagValue</u> and <u>NewTagValue</u> do not match.
711	Restricted object	<u>UpdateObject()</u> failed because the <u>@restricted</u> property of the object specified by the <u>ObjectID</u> argument is set to "1".
712	Bad metadata	<u>UpdateObject()</u> failed because the specified <u>Elements</u> argument is not supported or is invalid.
713	Restricted parent object	<u>UpdateObject()</u> failed because the <u>@restricted</u> property of the parent object of the object specified by the <u>ObjectID</u> argument is set to "1".

2.5.11 MoveObject()

This optional action moves ContentDirectory objects within the ContentDirectory service hierarchy when permitted. The caller specifies the ID of the object to move in the ObjectID input argument and the ID of the destination container in the NewParentID input argument and the action returns the object ID of the moved object after the move has completed in the NewObjectID output argument. The MoveObject() action may be invoked to move either containers or items. A container move action is a hierarchical move. If a container contains other objects, all contained objects must be moved along with the parent object. The object ID of the moved object or any of its descendent children MAY be changed by the move operation but all other object IDs MUST remain unchanged by the move operation. If a moved object is referenced by other objects, all references to the moved object must remain valid after the ContentDirectory service has completed the move operation. While implementers MAY choose to provide an implementation which changes the object ID of the objects being moved, this requirement may create a significant database problem for ContentDirectory service implementations with many entries. If a MoveObject() implementation changes the object IDs of moving objects, it MUST also send SystemUpdateID events and, if it supports the ContainerUpdateIDs state variable, it MUST send ContainerUpdateIDs events indicating which containers have changed. The ContainerUpdateIDs state variable MUST contain the object IDs of the old parent container and the new parent container.

If the NewObjectID output argument is identical to the ObjectID input argument, a control point can conclude that no object IDs changed during the execution of the MoveObject() action. That is, it is illegal, during a container move, to change the object ID of any contained object without also changing the object ID of the container that the action specified in the MoveObject() action.

The entire requested move MUST complete or it MUST fail and leave the ContentDirectory service hierarchy unchanged.

Browse() and Search() actions depend upon the presence of a coherent ContentDirectory service hierarchy. If a Browse() or Search() action is invoked by a control point while the MoveObject() action is executing, the ContentDirectory service implementation is responsible for coordinating ContentDirectory service operations so that control points receive coherent results.

- If the object to be moved is restricted (indicated by its @restricted property set to true), the action must fail with error code 711 (Restricted object).
- If the destination container is restricted (indicated by its @restricted property set to true), the action must fail with error code 713 (Restricted destination parent object).
- If the parent of the object to be moved is restricted (indicated by its @restricted property set to true), the action must fail with error code 721 (Restricted source parent object).
- The class of the object to be moved must be compatible with the upnp:createClass property of the destination container. If the class is not compatible, the action must fail with error code 722.
- If the move operation would create an illegal configuration for the ContentDirectory service hierarchy, the action must fail with error code 723 (Illegal move destination). This may happen, for example, if the requested destination container is a child of the container to be moved.

2.5.11.1 Arguments

Table 2-28: Arguments for MoveObject()

Argument	Direction	Related State Variable
<u>ObjectID</u>	<u>IN</u>	<u>A ARG TYPE ObjectID</u>
<u>NewParentID</u>	<u>IN</u>	<u>A ARG TYPE ObjectID</u>
<u>NewObjectID</u>	<u>OUT</u>	<u>A ARG TYPE ObjectID</u>

2.5.11.2 Dependency on State

None.

2.5.11.3 Effect on State

This action updates the SystemUpdateID state variable. This action also updates the ContainerUpdateID property for the destination parent container and the source parent Container. Also, the ContainerUpdateID property must be updated for each container that contains objects with references to objects whose object ID changed.

2.5.11.4 Errors

Table 2-29: Error Codes for MoveObject()

Error Code	Error Description	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	No such object	The object identified by <u>ObjectID</u> does not exist.
710	No such container	The container identified by <u>NewParentID</u> does not exist.
711	Restricted object	Cannot move the object because the object's <u>@restricted</u> property is set to " <u>I</u> ".
713	Restricted parent object	<u>MoveObject()</u> failed because the <u>@restricted</u> property of the destination parent container is set to " <u>I</u> ".
721	Restricted source parent object	<u>MoveObject()</u> failed because the <u>@restricted</u> property of the source parent container of the object to move is set to " <u>I</u> ".
722	Incompatible parent class	<u>MoveObject()</u> failed because the class of the object to move is not compatible with the <u>upnp:createClass</u> property of the destination parent container.
723	Illegal destination	<u>MoveObject()</u> failed because the specified move would create an illegal configuration.

2.5.12 ImportResource()

This action transfers a file from an external source, specified by the SourceURI argument, to a local destination in the ContentDirectory service, specified by the DestinationURI argument. The control point invokes the ImportResource() action with the SourceURI argument set to the URI of the external location and the DestinationURI argument set to the value of the res@importURI property associated with the destination object's res property. The ImportResource() action MUST use HTTP-GET on the SourceURI to retrieve the external content and to create a local copy of it.

The DestinationURI should correspond to an existing res@importURI property in the ContentDirectory service implementation. The res@importURI property identifies a download portal for the associated res property of a specific target object. It is used to create a local copy of the external content. After the transfer finishes successfully, the local content is then associated with the target object by setting the target object's res property value to a URI for that content, which MAY or MAY NOT be the same URI as the one specified in the res@importURI property, depending on the ContentDirectory service implementation. If the res property of the target object already has a value when the ImportResource() action is invoked, the resource is updated and the value of the res property MAY be changed.

When the ContentDirectory service validates the destination location in the ContentDirectory service implementation, the action returns a unique TransferID in the response and starts transferring the content. A control point can monitor the progress of the transfer by invoking the GetTransferProgress() action.

2.5.12.1 Arguments

Table 2-30: Arguments for ImportResource()

Argument	Direction	Related State Variable
<u>SourceURI</u>	<u>IN</u>	<u>A_ARG_TYPE_URI</u>
<u>DestinationURI</u>	<u>IN</u>	<u>A_ARG_TYPE_URI</u>
<u>TransferID</u>	<u>OUT</u>	<u>A_ARG_TYPE_TransferID</u>

2.5.12.2 Dependency on State

None.

2.5.12.3 Effect on State

When the file transfer is started, the TransferID value returned by the ImportResource() action is added into the TransferIDs state variable. When the file transfer is finished, the TransferID value is removed from the TransferIDs state variable.

2.5.12.4 Errors

Table 2-31: Error Codes for ImportResource()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
714	No such source resource	<u>ImportResource()</u> failed because the source resource specified by the <u>SourceURI</u> argument cannot be identified.
715	Source resource access denied	<u>ImportResource()</u> failed because the source resource specified by the <u>SourceURI</u> argument cannot be accessed.
716	Transfer busy	<u>ImportResource()</u> failed because the source resource specified by the <u>SourceURI</u> argument refuses to perform another file transfer.
718	No such destination resource	<u>ImportResource()</u> failed because the destination resource specified by the <u>DestinationURI</u> argument cannot be identified.
719	Destination resource access denied	<u>ImportResource()</u> failed because the destination resource specified by the <u>DestinationURI</u> argument cannot be accessed.

2.5.13 ExportResource()

This action transfers a file, using HTTP POST, from a local source, specified by the SourceURI input argument, to an external destination, specified by the DestinationURI input argument. When the ContentDirectory service validates the source location, the action returns a unique TransferID in the response and starts the HTTP POST. A control point can monitor the progress of the file transfer by using the GetTransferProgress() action. Note that the transfer does not remove the resource from the ContentDirectory service. The transfer simply copies the existing resource to an external destination.

2.5.13.1 Arguments

Table 2-32: Arguments for ExportResource()

Argument	Direction	Related State Variable
<u>SourceURI</u>	<u>IN</u>	<u>A_ARG_TYPE_URI</u>
<u>DestinationURI</u>	<u>IN</u>	<u>A_ARG_TYPE_URI</u>
<u>TransferID</u>	<u>OUT</u>	<u>A_ARG_TYPE_TransferID</u>

2.5.13.2 Dependency on State

None.

2.5.13.3 Effect on State

When the file transfer is started, the *TransferID* returned by *ExportResource()* is added into the TransferIDs state variable. When the file transfer is finished, TransferID is removed from the TransferIDs state variable.

2.5.13.4 Errors

Table 2-33: Error Codes for *ExportResource()*

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
714	No such source resource	<i>ExportResource()</i> failed because the source resource specified by the <i>SourceURI</i> argument cannot be identified.
715	Source resource access denied	<i>ExportResource()</i> failed because the source resource specified by the <i>SourceURI</i> argument cannot be accessed.
716	Transfer busy	<i>ExportResource()</i> failed because the source resource specified by the <i>SourceURI</i> argument refuses to perform another file transfer.
718	No such destination resource	<i>ExportResource()</i> failed because the destination resource specified by the <i>DestinationURI</i> argument cannot be identified.
719	Destination resource access denied	<i>ExportResource()</i> failed because the destination resource specified by the <i>DestinationURI</i> argument cannot be accessed.

2.5.14 *DeleteResource()*

This action uses the specified *ResourceURI* to locate all of the *res* properties whose value equals the value specified in the *ResourceURI* input argument in the ContentDirectory service, and then delete those *res* properties and all of their associated *res@xxx* properties from the respective objects. As a result, all located objects will end up with one less *res* property and in some cases some objects may end up without any *res* properties.

Whether or not the resource identified by *ResourceURI* is actually deleted is implementation dependent. For ContentDirectory service implementations that *do* attempt to delete resources identified by *ResourceURI*, there are three likely results of the *DeleteResource()* action:

- The *DeleteResource()* action returns successfully, indicating that the resource identified by *ResourceURI* was found and deleted.
- The *DeleteResource()* action fails and returns error code 714 indicating that an unsuccessful attempt was made to delete the resource identified by *ResourceURI* because the resource was not found. No resources are deleted and there is no change in state of the ContentDirectory service.
- The *DeleteResource()* action fails and returns error code 715 indicating that an unsuccessful attempt was made to delete the resource identified by *ResourceURI* because the resource could not be accessed. No resources are deleted and there is no change in state of the ContentDirectory service.

2.5.14.1 Arguments

Table 2-34: Arguments for *DeleteResource()*

Argument	Direction	Related State Variable
ResourceURI	IN	A_ARG_TYPE_URI

2.5.14.2 Dependency on State

None.

2.5.14.3 Effect on State

This action updates SystemUpdateID. Also updates ContainerUpdateId for each container in which a res property is removed from an object.

2.5.14.4 Errors

Table 2-35: Error Codes for DeleteResource()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
714	No such resource	<u>DeleteResource()</u> failed because the resource specified by <u>ResourceURI</u> argument was not found.
715	Source resource access denied	<u>DeleteResource()</u> failed because the resource specified by <u>ResourceURI</u> argument cannot be accessed.

2.5.15 StopTransferResource()

This action stops the file transfer initiated by the ImportResource() or ExportResource() action. The file transfer, identified by the TransferID argument, is halted immediately.

2.5.15.1 Arguments

Table 2-36: Arguments for StopTransferResource()

Argument	Direction	Related State Variable
<u>TransferID</u>	<u>IN</u>	<u>A_ARG_TYPE_TransferID</u>

2.5.15.2 Dependency on State

None.

2.5.15.3 Effect on State

When the file transfer is finished, TransferID is removed from the status TransferIDs.

2.5.15.4 Errors

Table 2-37: Error Codes for StopTransferResource()

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
717	No such file transfer	<u>StopTransferResource()</u> failed because the file transfer task specified by the <u>TransferID</u> argument does not exist.

2.5.16 GetTransferProgress()

This action is used to query the progress of the file transfer initiated by the ImportResource() or the ExportResource() action. Progress of the file transfer, specified by TransferID, will be returned in the response. The TransferStatus argument indicates the status of the file transfer. It can be either “IN_PROGRESS”, “STOPPED”, “ERROR”, or “COMPLETED”. The TransferLength argument specifies the length in bytes that has been transferred so far. The TransferTotal argument specifies the total length of the file in bytes that is expected to be transferred. If the ContentDirectory service cannot determine the total length, the TransferTotal argument MUST be set to zero. If the file transfer is started, the status is changed to “IN_PROGRESS”. If the file transfer is finished, the status is changed to either “STOPPED”, “ERROR”, or “COMPLETED” depending on the result of the file transfer. The ContentDirectory service MUST maintain the status of a file transfer for at least 30 seconds after the file transfer has finished allowing a control point to query the result of the file transfer.

2.5.16.1 Arguments

Table 2-38: Arguments for GetTransferProgress()

Argument	Direction	Related State Variable
<u>TransferID</u>	<u>IN</u>	<u>A_ARG_TYPE_TransferID</u>
<u>TransferStatus</u>	<u>OUT</u>	<u>A_ARG_TYPE_TransferStatus</u>
<u>TransferLength</u>	<u>OUT</u>	<u>A_ARG_TYPE_TransferLength</u>
<u>TransferTotal</u>	<u>OUT</u>	<u>A_ARG_TYPE_TransferTotal</u>

2.5.16.2 Dependency on State

None.

2.5.16.3 Effect on State

None.

2.5.16.4 Errors

Table 2-39: Error Codes for GetTransferProgress()

Error Code	Error Description	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
717	No such file transfer	<u>GetTransferProgress()</u> failed because the file transfer task specified by the <u>TransferID</u> argument does not exist.

2.5.17 CreateReference()

This action creates a reference to an existing item, specified by the ObjectID argument, in the parent container, specified by the ContainerID argument. Both the ContainerID and ObjectID MUST already exist in the ContentDirectory service. A unique, new object ID is assigned to the newly created *reference item* (in its @id property) and returned in the NewID output argument.

Refer to Section 2.2, “Terms” for detailed information about *reference items*.

2.5.17.1 Arguments

Table 2-40: Arguments for **CreateReference()**

Argument	Direction	Related State Variable
<u>ContainerID</u>	<u>IN</u>	<u>A_ARG_TYPE_ObjectID</u>
<u>ObjectID</u>	<u>IN</u>	<u>A_ARG_TYPE_ObjectID</u>
<u>NewID</u>	<u>OUT</u>	<u>A_ARG_TYPE_ObjectID</u>

2.5.17.2 Dependency on State

None.

2.5.17.3 Effect on State

A new reference object is added to the specified parent object.

2.5.17.4 Errors

Table 2-41: Error Codes for **CreateReference()**

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	No such object	CreateReference() failed because the specified <u>ObjectID</u> argument is invalid.
710	No such container	CreateReference() failed because the <u>ContainerID</u> argument is invalid or identifies an object that is not a container.
713	Restricted parent object	CreateReference() failed because the <u>@restricted</u> property of the parent object of the object specified by the <u>ObjectID</u> argument is set to " <u>I</u> ".

2.5.18 Non-Standard Actions Implemented by a UPnP Vendor

To facilitate certification, non-standard actions implemented by a UPnP vendor MUST be included in the device's service template. The UPnP Device Architecture lists naming requirements for non-standard actions (cfr. Section on Description).

2.5.19 Common Error Codes

The following table lists error codes common to actions for this service type. If a given action results in multiple errors, the most specific error **MUST** be returned.

Table 2-42: Common error codes

errorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	No such object	The action failed because a specified object is invalid.
702	Invalid CurrentTagValue	The action failed because a specified tag/value pair does not match the current state of the ContentDirectory service.
703	Invalid NewTagValue	The action failed because the specified tag value is invalid.
704	Required tag	The action failed because the request included an implicit request to delete a required tag.
705	Read only tag	The action failed because the request included an implicit request to modify a read-only tag.
706	Parameter Mismatch	The action failed because two separate references to the number of tag/value pairs (including empty placeholders) do not match.
708	Unsupported or invalid search criteria	The action failed because a specified search criteria argument is not supported or is invalid.
709	Unsupported or invalid sort criteria	The action failed because a specified sort criteria argument is not supported or is invalid.
710	No such container	The action failed because an argument specifying a container is invalid or identifies an object that is not a container.
711	Restricted object	The action failed because it would result in the modification of a restricted object.
712	Bad metadata	The action failed because a specified XML tag is not supported or because a specified <i>DIDL-Lite XML Document or Fragment</i> is invalid.
713	Restricted parent object	The action failed because it would result in the modification of the restricted parent object of the target object.
714	No such source resource	The action failed because a specified source resource was not found.
715	Source resource access denied	The action failed because a specified source resource is busy.
716	Transfer busy	The action failed because a specified resource refuses to perform another file transfer.
717	No such file transfer	The action failed because a specified file transfer task does not exist.
718	No such destination resource	The action failed because a specified destination resource cannot be identified.

errorCode	errorDescription	Description
719	Destination resource access denied	The action failed because a specified destination resource is busy.
720	Cannot process the request	The action failed because the ContentDirectory service was unable to complete the necessary computations in the time allotted.
721	Restricted source parent object	The action failed because the <u>@restricted</u> property of the source parent container of the object to move is set to <i>true</i> .
722	Incompatible parent class	The action failed because the class of the object to move is not compatible with the <u>upnp:createClass</u> property of the destination parent container.
723	Illegal destination	The action failed because it would create an illegal configuration.

Note 1: The errorDescription field returned by an action does not necessarily contain human-readable text (for example, as indicated in the second column of the Error Code tables.) It may contain machine-readable information that provides more detailed information about the error. It is therefore not advisable for a control point to blindly display the errorDescription field contents to the user.

Note 2: 800-899 Error Codes are not permitted for standard actions. See UPnP Device Architecture section on Control for more details.

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008

2.6 Theory of Operation (Informative)

2.6.1 Introduction

This section walks through several scenarios to illustrate the various actions supported by the ContentDirectory service. These include browsing, searching, object creation, update, and deletion, property creation, update and deletion, content transfer, playlist manipulation, Internet content representation, and bookmark manipulation.

2.6.2 Content Setup for Browsing and Searching

The following illustrates the logical structure of a ContentDirectory service which exposes a physical directory structure on a PC-like file system. The content includes music and photos organized into a few directory folders. The logical directory hierarchy is as follows:

- Name="Content"
 - Name="My Music"
 - Name="Singles Soundtrack - Various Artists.musicalbum"
 - Name="Would - Alice In Chains.wma", Size="90000"
 - Name="Chloe Dancer - Mother Love Bone.wma", Size="200000"
 - Name="State Of Love And Trust - Pearl Jam.wma", Size="70000"
 - Name="Drown - Smashing Pumpkins.mp3", Size="140000"
 - Name="Brand New Day - Sting.musicalbum"
 - Name="A Thousand Years - Sting.wma", Size="100000"
 - Name="Desert Rose - Sting.wma", Size="50000"
 - Name="Big Lie Small World - Sting.mp3", Size="80000"
 - Name="My Photos"
 - Name="Mexico Trip.photoalbum"
 - Name="Sunset on the beach - 10/20/2001.jpg", Size="20000"
 - Name="Playing in the pool - 10/25/2001.jpg", Size="25000"
 - Name="Christmas.photoalbum"
 - Name="John and Mary by the fire - 12/24/2001.jpg", Size="22000"
 - Name="Christmas Tree loaded with presents - 12/25/2001.jpg", Size="10000"
 - Name="Album Art"
 - Name="Brand New Day.albumart", Size="20000"
 - Name="Singles Soundtrack.albumart", Size="20000"

2.6.3 Browsing

The [*Browse\(\)*](#) action is designed to allow the control point to navigate the *native* content hierarchy exposed by the ContentDirectory service. This hierarchy could map onto an explicit physical hierarchy or a logical one. In addition, the [*Browse\(\)*](#) action enables the following features while navigating the hierarchy:

- **Metadata only browsing.** The metadata associated with a particular object can be retrieved.
- **Children object browsing.** The direct children of an object whose class is derived from the container class can be retrieved.
- **Incremental navigation** that is: the full hierarchy is never returned in one action since this is likely to flood the resources available to the control point (memory, network bandwidth, etc.). Also within a particular hierarchy level, the control point can restrict the number (and the starting offset) of objects returned in the result.

- **Sorting.** The result can be requested in a particular sort order. The available sort orders are expressed in the return value of the *GetSortCapabilities()* action.
- **Filtering.** The result data can be filtered to only include a subset of the properties available on the object (see Section 2.3.13, “*A ARG TYPE Filter*.”) Note that certain properties MUST NOT be filtered out in order to maintain validity of the resulting *DIDL-Lite XML Document*. If a non-filterable property is left out of the *filter* list, it will still be included in the *Result* argument.

The following examples illustrate the typical *Browse()* request-response interaction between a control point and a ContentDirectory service. It assumes the content setup specified in Section 2.6.2, “Content Setup for Browsing and Searching.”

2.6.3.1 Retrieving Sort Capabilities

When it connects to the ContentDirectory service, the control point determines which properties can be used as sort criteria in a *Browse()* or *Search()* request. It does this via the *GetSortCapabilities* action:

Request:

```
GetSortCapabilities()
```

Response:

```
GetSortCapabilities("dc:title,dc:creator,dc:date,didl-lite:res@size")
```

2.6.3.2 Browsing the Root Level Metadata

The control point needs to retrieve the root level metadata for the ContentDirectory service. It does this via the following *Browse()* action:

Request:

```
Browse("0", "BrowseMetadata", "*", 0, 0, "")
```

Response:

```
Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <container id="0" parentID="-1" childCount="3"
    restricted="1" searchable="1">
    <dc:title>Content</dc:title>
    <upnp:class>object.container.storageFolder</upnp:class>
    <upnp:storageUsed>847000</upnp:storageUsed>
    <upnp:writeStatus>WRITABLE</upnp:writeStatus>
    <upnp:searchClass includeDerived="0">
      object.container.album.musicAlbum
    </upnp:searchClass>
    <upnp:searchClass includeDerived="0">
      object.container.album.photoAlbum
    </upnp:searchClass>
    <upnp:searchClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:searchClass>
    <upnp:searchClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:searchClass>
    <upnp:searchClass name="Vendor Album Art"
      includeDerived="1">
```

```

        object.item.imageItem.photo.vendorAlbumArt
    </upnp:searchClass>
</container>
</DIDL-Lite>", 1, 1, 10)

```

Note that the response contains the *DIDL-Lite XML Document* with the metadata corresponding to the root container of the ContentDirectory service (container @id = 0), and the other output arguments NumberReturned, TotalMatches, and ContainerUpdateID, respectively.

2.6.3.3 Browsing the Children of the Root Level

The control point needs to retrieve the children of the root-level container. The control point can display 3 items at a time, so it restricts the number of children returned in the Result argument. It does this via the following Browse() action:

Request:

```
Browse("0", "BrowseDirectChildren", "*", 0, 3, "")
```

Response:

```

Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
      http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
      http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <container id="1" parentID="0" childCount="2" restricted="0">
    <dc:title>My Music</dc:title>
    <upnp:class>object.container.storageFolder</upnp:class>
    <upnp:storageUsed>730000</upnp:storageUsed>
    <upnp:writeStatus>WRITABLE</upnp:writeStatus>
    <upnp:searchClass includeDerived="0">
      object.container.album.musicAlbum
    </upnp:searchClass>
    <upnp:searchClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.container.album.musicAlbum
    </upnp:createClass>
  </container>
  <container id="2" parentID="0" childCount="2" restricted="0">
    <dc:title>My Photos</dc:title>
    <upnp:class>object.container.storageFolder</upnp:class>
    <upnp:storageUsed>77000</upnp:storageUsed>
    <upnp:writeStatus>WRITABLE</upnp:writeStatus>
    <upnp:searchClass includeDerived="0">
      object.container.album.photoAlbum
    </upnp:searchClass>
    <upnp:searchClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.container.album.photoAlbum
    </upnp:createClass>
  </container>
  <container id="30" parentID="0" childCount="2" restricted="0">
    <dc:title>Album Art</dc:title>

```

```
<upnp:class>object.container.storageFolder</upnp:class>
<upnp:storageUsed>40000</upnp:storageUsed>
<upnp:writeStatus>WRITABLE</upnp:writeStatus>
<upnp:searchClass name="Vendor Album Art"
  includeDerived="1">
  object.item.imageItem.photo.vendorAlbumArt
</upnp:searchClass>
<upnp:createClass includeDerived="1">
  object.item.imageItem.photo.vendorAlbumArt
</upnp:createClass>
</container>
</DIDL-Lite> ", 3, 3, 10)
```

2.6.3.4 Browsing the Children of the My Music Folder

The control point needs to retrieve the children of the My Music folder. The control point can display 3 items at a time, so it specifies the number of children returned in the [Result](#) argument. In addition, it specifies the [Result](#) argument to be sorted in ascending order by the [creator](#) property. It does this via the following [Browse\(\)](#) action:

Request :

```
Browse("1", "BrowseDirectChildren", "*", 0, 3, "+dc:creator")
```

Response :

```
Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <container id="4" parentID="1" childCount="3" restricted="0">
    <dc:title>Brand New Day</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>object.container.album.musicAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:createClass>
  </container>
  <container id="3" parentID="1" childCount="4" restricted="0">
    <dc:title>Singles Soundtrack</dc:title>
    <dc:creator>Various Artists</dc:creator>
    <upnp:class>object.container.album.musicAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:createClass>
  </container>
</DIDL-Lite> ", 2, 2, 21)
```

2.6.3.5 Browsing the Children of the Singles Soundtrack Music Album

The control point needs to retrieve the children of the Singles Soundtrack music album. The control point can display 3 items at a time, so it restricts the number of children returned in each [Result](#) argument. In addition, it

specifies the Result argument to be sorted in ascending order by the dc:title property. It does this via the following Browse() action:

Request:

```
Browse("3", "BrowseDirectChildren", "*", 0, 3, "+dc:title")
```

Response:

```
Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="6" parentID="3" restricted="0">
    <dc:title>Chloe Dancer</dc:title>
    <dc:creator>Mother Love Bone</dc:creator>
    <upnp:class>object.item.audioItem.musicTrack</upnp:class>
    <res protocolInfo="http-get:*:audio/x-ms-wma:*" size="200000">
      http://10.0.0.1/getcontent.asp?id=6
    </res>
  </item>
  <item id="8" parentID="3" restricted="0">
    <dc:title>Drown</dc:title>
    <dc:creator>Smashing Pumpkins</dc:creator>
    <upnp:class>object.item.audioItem.musicTrack</upnp:class>
    <res protocolInfo="http-get:*:audio/mpeg:*" size="140000">
      http://10.0.0.1/getcontent.asp?id=8
    </res>
  </item>
  <item id="7" parentID="3" restricted="0">
    <dc:title>State Of Love And Trust</dc:title>
    <dc:creator>Pearl Jam</dc:creator>
    <upnp:class>object.item.audioItem.musicTrack</upnp:class>
    <res protocolInfo="http-get:*:audio/x-ms-wma:*" size="70000">
      http://10.0.0.1/getcontent.asp?id=7
    </res>
  </item>
</DIDL-Lite>", 3, 4, 18)
```

Request:

```
Browse("3", "BrowseDirectChildren", "*", 3, 3, "+dc:title")
```

Response:

```
Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="5" parentID="3" restricted="0">
    <dc:title>Would</dc:title>
```

```

<dc:creator>Alice In Chains</dc:creator>
<upnp:class>object.item.audioItem.musicTrack</upnp:class>
<res protocolInfo="http-get:*:audio/x-ms-wma:*" size="90000">
  http://10.0.0.1/getcontent.asp?id=5
</res>
</item>
</DIDL-Lite>", 1, 4, 18)

```

2.6.3.6 Browsing the Children of the Album Art Folder

The control point needs to retrieve the children of the Album Art folder. The control point can display 3 items at a time so it restricts the number of children returned in the [Result](#) argument. It does this via the following [Browse\(\)](#) action:

Request:

```
Browse("30", "BrowseDirectChildren", "*", 0, 3, "")
```

Response:

```

Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="31" parentID="30" restricted="0">
    <dc:title>Brand New Day</dc:title>
    <upnp:class name="Vendor Album Art">
      object.item.imageItem.photo.vendorAlbumArt
    </upnp:class>
    <res protocolInfo="http-get:*:image/jpeg:*" size="20000">
      http://10.0.0.1/getcontent.asp?id=31
    </res>
  </item>
  <item id="32" parentID="30" restricted="0">
    <dc:title>Singles Soundtrack</dc:title>
    <upnp:class name="Vendor Album Art">
      object.item.imageItem.photo.vendorAlbumArt
    </upnp:class>
    <res protocolInfo="http-get:*:image/jpeg:*" size="20000">
      http://10.0.0.1/getcontent.asp?id=32
    </res>
  </item>
</DIDL-Lite>", 2, 2, 50)

```

2.6.4 Searching

The [Search\(\)](#) action is designed to allow a control point to search for objects in the ContentDirectory service that match a given search criteria (see Section 2.3.11, "[A_ARG_TYPE_SearchCriteria](#)."). In addition, the [Search\(\)](#) action supports the following features:

- **Incremental result retrieval** that is: in the context of a particular request the control point can restrict the number (and the starting offset) of objects returned in the [Result](#) argument.
- **Sorting.** The [Result](#) can be requested in a particular sort order. The available sort orders are expressed in the return value of the [GetSortCapabilities](#) action.
- **Filtering.** The [Result](#) data can be filtered to only include a subset of the properties available on the object (see Section 2.3.13, "[A_ARG_TYPE_Filter](#)."). Note that certain properties MUST NOT be

filtered out in order to maintain the validity of the resulting *DIDL-Lite XML Document*. If a non-filterable property is left out of the filter set, it will still be included in the [Result](#) argument.

The following examples illustrate the typical [Search\(\)](#) request-response interaction between a control point and a ContentDirectory service. It assumes the content setup specified in Section 2.6.2, “Content Setup for Browsing and Searching.”

2.6.4.1 Retrieving Search Capabilities

When it connects to the ContentDirectory service, the control point determines which properties can be used in the [SearchCriteria](#) argument of the [Search\(\)](#) action. It does this via the [GetSearchCapabilities\(\)](#) action:

Request:

```
GetSearchCapabilities()
```

Response:

```
GetSearchCapabilities("
dc:title,dc:creator,dc:date,upnp:class,res@size")
```

2.6.4.2 Search for All Content Created by the performer Sting

Search for all objects where [dc:creator](#) is *Sting* and sort the [Result](#) argument in ascending order by [dc:title](#). The control point can only display 3 items at a time so it restricts the number requested. The following [Search\(\)](#) action is used:

Request:

```
Search("0", "dc:creator = \"Sting\"", "*", 0, 3, "+dc:title")
```

Response:

```
Search("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
      http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
      http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="9" parentID="4" restricted="0">
    <dc:title>A Thousand Years</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>object.item.audioItem.musicTrack</upnp:class>
    <res protocolInfo="http-get:*:audio/x-ms-wma:*" size="100000">
      http://10.0.0.1/getcontent.asp?id=9
    </res>
  </item>
  <item id="11" parentID="4" restricted="0">
    <dc:title>Big Lie, Small World</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>object.item.audioItem.musicTrack</upnp:class>
    <res protocolInfo="http-get:*:audio/mpeg:*" size="70000">
      http://10.0.0.1/getcontent.asp?id=11
    </res>
  </item>
  <container id="4" parentID="1" childCount="3" restricted="0"
    searchable="1">
    <dc:title>Brand New Day</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>object.container.album.musicAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.audioItem.musicTrack
```

```

        </upnp:searchClass>
        <upnp:createClass includeDerived="0">
            object.item.audioItem.musicTrack
        </upnp:createClass>
    </container>
</DIDL-Lite>", 3, 4, 10)

```

Request:

```
Search("0", "dc:creator = \"Sting\"", "*", 3, 3, "+dc:title")
```

Response:

```

Search("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="10" parentID="4" restricted="0">
    <dc:title>Desert Rose</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>object.item.audioItem.musicTrack</upnp:class>
    <res protocolInfo="http-get:*:audio/x-ms-wma:*" size="50000">
      http://10.0.0.1/getcontent.asp?id=10
    </res>
  </item>
</DIDL-Lite>", 1, 4, 10)

```

2.6.4.3 Search for all Photos Taken During the Month of October

Search for all photo objects whose dc:date is in October and sort the Result argument in ascending order by dc:date. The control point can only display 3 items at a time so it restricts the number requested. The following Search() action is used:

Request:

```

Search("0",
"upnp:class derivedfrom \"object.item.imageItem.photo\" and (dc:date >=
\"2001-10-01\" and dc:date <= \"2001-10-31\")", "*", 0, 3, "+dc:date")

```

Response:

```

Search("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="14" parentID="12" restricted="0">
    <dc:title>Sunset on the beach</dc:title>
    <dc:date>2001-10-20</dc:date>
    <upnp:class>object.item.imageItem.photo</upnp:class>
    <res protocolInfo="http-get:*:image/jpeg:*" size="20000">
      http://10.0.0.1/getcontent.asp?id=14
    </res>
  </item>
</DIDL-Lite>

```

```

</item>
<item id="15" parentID="12" restricted="0">
  <dc:title>Playing in the pool</dc:title>
  <dc:date>2001-10-25</dc:date>
  <upnp:class>object.item.imageItem.photo</upnp:class>
  <res protocolInfo="http-get:*:image/jpeg:*" size="25000">
    http://10.0.0.1/getcontent.asp?id=15
  </res>
</item>
</DIDL-Lite>", 2, 2, 10)

```

2.6.4.4 Search for All Objects in the My Photos Folder Containing the Word “Christmas”

Search for all objects where the title contains “Christmas” under the My Photos folder. The control point can only display 3 items at a time so it restricts the number requested. The [Result](#) argument is sorted in ascending order by [dc:title](#). The following [Search\(\)](#) action is used:

Request:

```
Search("2", "dc:title contains \"Christmas\"", "*", 0, 3, "+dc:title")
```

Response:

```

Search("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <container id="13" parentID="2" restricted="0"
    searchable="1">
    <dc:title>Christmas</dc:title>
    <upnp:class>object.container.album.photoAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:createClass>
  </container>
  <item id="17" parentID="13" restricted="0">
    <dc:title>Christmas tree loaded with presents</dc:title>
    <dc:date>2001-12-25</dc:date>
    <upnp:class>object.item.imageItem.photo</upnp:class>
    <res protocolInfo="http-get:*:image/jpeg:*" size="25000">
      http://10.0.0.1/getcontent.asp?id=17
    </res>
  </item>
</DIDL-Lite>", 2, 2, 47)

```

2.6.4.5 Search for all [album](#) objects in the ContentDirectory service

Search for all objects that are derived from [object.container.album](#). The following [Search\(\)](#) action is used:

Request:

```
Search("0", "upnp:class derivedfrom \"object.container.album\"", "*", 0, 4,
  "")
```

Response:

```

Search("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <container id="3" parentID="1" childCount="4" restricted="0"
    searchable="1">
    <dc:title>Singles Soundtrack</dc:title>
    <dc:creator>Various Artists</dc:creator>
    <upnp:class>object.container.album.musicAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:createClass>
  </container>
  <container id="4" parentID="1" childCount="3" restricted="0"
    searchable="1">
    <dc:title>Brand New Day</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>object.container.album.musicAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.item.audioItem.musicTrack
    </upnp:createClass>
  </container>
  <container id="12" parentID="2" restricted="0"
    searchable="1">
    <dc:title>Mexico Trip</dc:title>
    <upnp:class>object.container.album.photoAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:createClass>
  </container>
  <container id="13" parentID="2" restricted="0"
    searchable="1">
    <dc:title>Christmas</dc:title>
    <upnp:class>object.container.album.photoAlbum</upnp:class>
    <upnp:searchClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:searchClass>
    <upnp:createClass includeDerived="0">
      object.item.imageItem.photo
    </upnp:createClass>
  </container>
</DIDL-Lite>", 4, 4, 10)

```

2.6.5 Browsing, Searching, and References

Using the content setup above, the following examples illustrate creation of a reference, the result of a search where the result contains a reference, and deletion of a reference.

2.6.5.1 Creating a reference to a photo in the Mexico Trip album inside the Christmas album

A reference to an existing item is created via the following action:

Request :

```
CreateReference("13", "15")
```

Response :

```
CreateReference("20")
```

2.6.5.2 Search for All Photos Taken During the Month of October

Search for all photo objects whose dc:date is in October and sort the Result argument in ascending order by dc:date. The control point can only display 3 items at a time so it restricts the number requested. The following Search() action is used:

Request :

```
Search("0",
"upnp:class derivedfrom \"object.item.imageItem.photo\" and (dc:date >=
\"2001-10-01\" and dc:date <= \"2001-10-31\")", "*", 0, 3, "+dc:date")
```

Response :

```
Search("
<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<DIDL-Lite
  xmlns:dc=\"http://purl.org/dc/elements/1.1/\"
  xmlns=\"urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/\"
  xmlns:upnp=\"urn:schemas-upnp-org:metadata-1-0/upnp/\"
  xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
  xsi:schemaLocation=
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="14" parentID="12" restricted="0">
    <dc:title>Sunset on the beach</dc:title>
    <dc:date>2001-10-20</dc:date>
    <upnp:class>object.item.imageItem.photo</upnp:class>
    <res protocolInfo="http-get:*:image/jpeg:*" size="20000">
      http://10.0.0.1/getcontent.asp?id=14
    </res>
  </item>
  <item id="15" parentID="12" restricted="0">
    <dc:title>Playing in the pool</dc:title>
    <dc:date>2001-10-25</dc:date>
    <upnp:class>object.item.imageItem.photo</upnp:class>
    <res protocolInfo="http-get:*:image/jpeg:*" size="25000">
      http://10.0.0.1/getcontent.asp?id=15
    </res>
  </item>
```

```
<item id="20" refID="15" parentID="13" restricted="0">
  <dc:title>Playing in the pool</dc:title>
  <dc:date>2001-10-25</dc:date>
  <upnp:class>object.item.imageItem.photo</upnp:class>
  <res protocolInfo="http-get:*:image/jpeg:*" size="25000">
    http://10.0.0.1/getcontent.asp?id=15
  </res>
</item>
</DIDL-Lite>", 3, 3, 10)
```

2.6.5.3 Deletion of the Reference to the Photo in the Mexico Trip Album

A *reference item* is deleted via the [DestroyObject\(\)](#) action:

Request:

```
DestroyObject("20")
```

Response:

```
DestroyObject()
```

2.6.6 Object Creation

2.6.6.1 Creating a New Object

The [CreateObject\(\)](#) action is used to create a new object in the specified container. The ContentDirectory service will create an object according to the specified metadata. Additional metadata MAY be added by the ContentDirectory service. The action returns [ObjectID](#) and metadata of the created object. Note that all REQUIRED elements MUST exist in the returned [Result](#) argument.

2.6.6.2 Example: Creating a New MusicTrack in the Album1 (@id = 10)

Invoke [CreateObject\(\)](#) with the [ContainerID](#) argument set to 10 and the [Elements](#) argument set to the metadata describing the new object to be created. This must include the [upnp:class](#) property, and in this example, its value is set to "[object.item.audioItem.musicTrack](#)".

Request:

```
CreateObject("10", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
      http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
      http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="" parentID="10" restricted="0">
    <dc:title>New Track</dc:title>
    <upnp:class>
      object.item.audioItem.musicTrack
    </upnp:class>
  </item>
</DIDL-Lite>")
```

Response :

```

CreateObject ("12", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="12" parentID="10" restricted="0">
    <dc:title>New Track</dc:title>
    <dc:creator></dc:creator>
    <res importUri="http://pc/item?id=12"
      protocolInfo="*:*:audio:*">
    </res>
    <upnp:class>
      object.item.audioItem.musicTrack
    </upnp:class>
    <upnp:genre></upnp:genre>
    <upnp:album>Album1</upnp:album>
  </item>
</DIDL-Lite>")

```

2.6.7 Object Resource Binding (Importing a Resource)

There are two ContentDirectory service mechanisms defined to import content into the ContentDirectory service:

- The [*ImportResource\(\)*](#) action, which uses HTTP GET and the [*res@ImportUri*](#) property.
- HTTP POST, executed by the control point.

2.6.7.1 Transfer Using the [*ImportResource\(\)*](#) Action

The destination (for example <http://10.0.0.1/cd/import?id=3>) is located in the ContentDirectory service and the source that needs to be imported (for example <http://server/song.mp3>) is external to the ContentDirectory service. (Any resource identified by a URL can be used). If a control point wants to create a new object whose resource needs to be imported from an external source, it can first invoke the [*CreateObject\(\)*](#) action and then import the file.

After the [*CreateObject\(\)*](#) action, the [*res*](#) property of the newly created object holds the following value:

```

<res protocolInfo="*:*:audio:*"
  importUri="http://10.0.0.1/cd/import?id=3">
</res>

```

A control point then invokes the [*ImportResource\(\)*](#) action and a [*TransferID*](#) value (for example “1234”) is returned. The [*TransferID*](#) can be used by the control point to manipulate the transfer while it is progressing.

Request :

```

ImportResource ("http://server/song.mp3",
"http://10.0.0.1/cd/import?id=3")

```

Response :

```

ImportResource ("1234")

```

The ContentDirectory service initiates the HTTP GET to the external source and begins receiving data, which is directed to the local destination.

Request :

GET /song.mp3 HTTP/1.1

Response :

HTTP/1.1 200 OK

The control point may monitor the progress of the transfer using the GetTransferProgress() action:

Request :

GetTransferProgress ("1234")

Response :

GetTransferProgress ("IN_PROGRESS", 43852, 125327)

After the HTTP GET has finished successfully, the control point can query the result of the file transfer:

Request :

GetTransferProgress ("1234")

Response :

GetTransferProgress ("COMPLETED", 125327, 125327)

If a control point has subscribed to events from the ContentDirectory service, the control point receives two events from the TransferIDs state variable during the transfer described above:

The following event is generated when the actual transfer starts:

Event :

TransferIDs="1234"

When the transfer ends (either successfully or when it fails due to an error or is stopped by the StopTransferResource() action) a second event is generated:

Event :

TransferIDs=""

After the file transfer has completed successfully, the res property of the newly created object contains the following value (as an example):

```
<res protocolInfo="http-get:*:audio/m3u:*">
  http://10.0.0.1/cd/content?id=3
</res>
```

2.6.7.2 Transfer Using Direct HTTP POST

When the control point has direct access to the content (such as when the content is local to the control point), it is possible for a control point to post the desired content directly to the ContentDirectory service.

A control point initiates HTTP POST to the destination and begins sending the data.

Request :

POST /cd/content?id=3 HTTP/1.1

Response :

HTTP/1.1 200 OK

2.6.8 Exporting ContentDirectory Resources

There are two ContentDirectory service mechanisms defined to export content from the ContentDirectory service:

- The ExportResource() action.
- HTTP GET executed by the control point (only for resources that have the HTTP GET protocol specified in their res@protocolInfo property).

2.6.8.1 Transfer Using the ExportResource() Action

The source (for example `http://10.0.0.1/cd/content?id=3`) is located internal to the ContentDirectory service and the destination (for example `http://server/content?id=6`) is located externally and is identified by a URL.

For example, the res property of a ContentDirectory object contains the following value before the export:

```
<res protocolInfo="http-get:*:audio/m3u:*">
  http://10.0.0.1/cd/content?id=3
</res>
```

A control point invokes the ExportResource() action and a TransferID value (for example "1235") is returned:

Request :

```
ExportResource (
  "http://10.0.0.1/cd/content?id=3", "http://server/content?id=6")
```

Response :

```
ExportResource ("1235")
```

The ContentDirectory service initiates the HTTP POST to the external destination and begins sending data from the local source.

Request :

```
POST content?id=6 HTTP/1.1
```

Response :

```
HTTP/1.1 200 OK
```

The control point may monitor the progress of the transfer using the GetTransferProgress() action:

Request :

```
GetTransferProgress ("1235")
```

Response :

```
GetTransferProgress ("IN_PROGRESS", 43852, 125327)
```

After the HTTP POST has finished successfully, the control point can query the result of the file transfer:

Request :

```
GetTransferProgress ("1235")
```

Response :

```
GetTransferProgress ("COMPLETED", 125327, 125327)
```

If a control point has subscribed to events from the ContentDirectory service, the control point receives two events from the TransferIDs state variable during the transfer described above:

The following event is generated when the actual transfer starts:

Event :

```
TransferIDs="1235"
```

When the transfer ends (either successfully or when it fails due to an error or is stopped by the StopTransferResource() action) a second event is generated:

Event :

```
TransferIDs=""
```

After the file transfer has completed successfully, the res property of the object that contains the source, is unaltered:

```
<res protocolInfo="http-get:*:audio/m3u:*">
  http://10.0.0.1/cd/content?id=3
</res>
```

2.6.8.2 Transfer using HTTP GET

For any resource that supports the HTTP GET protocol as specified in its [res@protocolInfo](#) property, a control point initiates the transfer at the remote source, using HTTP GET. The resource is then copied from the ContentDirectory service to the control point.

2.6.9 Playlist Manipulation

2.6.9.1 Playlist File Representation in the ContentDirectory Service

A playlist file is represented as an object of the [playlist](#) class ([object.item.playlist](#)). The format of the playlist is indicated by the MIME type section of the [res@protocolInfo](#) property on the [playlist](#) object. If a search were performed for all objects of class [object.item.playlist](#) in the ContentDirectory service, it would return a [Result](#) of the following form:

```
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="2" parentID="1" restricted="0">
    <dc:title>Playlist of John and Mary's music</dc:title>
    <dc:creator>John Jones</dc:creator>
    <upnp:class>object.item.playlistItem</upnp:class>
    <res protocolInfo="http-get:*:audio/m3u:">
      http://pc/k.m3u
    </res>
  </item>
</DIDL-Lite>
```

2.6.9.2 Playlist File Generation

Objects derived from the [container](#) class ([object.container](#)) MAY contain objects derived from the [item](#) ([object.item](#)) or [container](#) classes. An example of such a class is the [musicAlbum](#) class ([object.container.album.musicAlbum](#)). It is desired to allow a control point to set up a rendering session of all the items in the music album. This may be accomplished by having the container object expose a [res](#) property, whose value is the URI of a playlist file in a format that is understood by the MediaRenderer. The content of the playlist file is a sequence of individual content items. Its internal format is identified by the [res@protocolInfo](#) property. Note: the order of the items in the playlist file is defined by the generator of the playlist, but SHOULD match the order of the items as returned from the [Browse\(\)](#) action on that container. The following example illustrates this:

- A [Browse\(\)](#) of a [musicAlbum](#) object's metadata returns a [Result](#) of the following form:

```
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <container id="1" parentID="0" restricted="0"
    searchable="1">
    <dc:title>Brand New Day</dc:title>
```

```

<dc:creator>Sting</dc:creator>
<upnp:class>
  object.container.album.musicAlbum
</upnp:class>
<upnp:searchClass includeDerived="0">
  object.item.audioItem.musicTrack
</upnp:searchClass>
<upnp:createClass includeDerived="0">
  object.item.audioItem.musicTrack
</upnp:createClass>
<res protocolInfo="http-get:*:audio/m3u:*">
  http://pc/genm3u?containerID="1"
</res>
</container>
</DIDL-Lite>

```

- A [Browse\(\)](#) of that [musicAlbum](#) object's direct children returns a [Result](#) of the following form:

```

<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
      http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
      http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="2" parentID="1" restricted="0">
    <dc:title>A Thousand Years</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>
      object.item.audioItem.musicTrack
    </upnp:class>
    <res protocolInfo="http-get:*:audio/x-ms-wma:*">
      http://pc/getcontent?contentID="2"
    </res>
  </item>
  <item id="3" parentID="1" restricted="0">
    <dc:title>Desert Rose</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>
      object.item.audioItem.musicTrack
    </upnp:class>
    <res protocolInfo="http-get:*:audio/x-ms-wma:*">
      http://pc/getcontent?contentID="3"
    </res>
  </item>
</DIDL-Lite>

```

- The control point uses the content of the [res](#) property on the [musicAlbum](#) container object in the [AVTransport::SetAVTransportURI\(\)](#) action on the MediaRenderer. The MediaRenderer then issues an HTTP GET on the URI "http://pc/genm3u?containerID="1"" to retrieve the generated M3U resource with the following content:

```

http://pc/getcontent?contentID="2"
http://pc/getcontent?contentID="3"

```

2.6.10 Internet Content Representation

A ContentDirectory service implementation will always reside on a UPnP device. However, various URIs present as metadata inside the ContentDirectory service are allowed to point to locations, for example, web

servers, that are outside the UPnP network. For example, an Internet Radio station may be represented by an object in a ContentDirectory service hosted by a UPnP MediaServer device.

In order to be compatible with as many renderer (player) devices in the UPnP home network as possible, a MediaServer device MAY be able to perform protocol and/or format conversion of content. Protocol and format information is exposed via the *res* and *res@protocolInfo* properties. MediaServer devices that can serve content using multiple protocols will generally have multiple *res* properties for a single object. For example, consider an Internet video resource using RTSP/RTP/UDP. To accommodate MediaRenderer devices that can only play via HTTP, a MediaServer could provide protocol translation, and offer the following metadata:

```
<item id="InternetStream1" restricted="0">
  <dc:title>Some Stream</dc:title>
  <upnp:class>
    object.item.videoItem
  </upnp:class>
  <res protocolInfo="rtsp-rtp-udp:*:MPV:*">
    rtsp://internet-server/stream1.m2v
  </res>
  <res protocolInfo="http-get:*:video/mpeg:*">
    http://upnp-device/stream1.m2v
  </res>
</item>
```

MediaRenderer devices that can deal with RTSP/RTP/UDP streams can play from the Internet server directly, whereas MediaRenderer devices that can only deal with HTTP streams would stream the same content over HTTP via the MediaServer device that acts as a translating proxy.

2.6.11 Bookmark Manipulation

The *CreateObject()*, *Browse()*, and *DestroyObject()* actions are used to manipulate bookmark objects.

2.6.11.1 *bookmarkItem* Example

The following is an example of a *bookmarkItem*.

```
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="bookmark-763215" parentID="BC_001"
    restricted="0">
    <dc:title>Gone with the Wind</dc:title>
    <upnp:class>object.item.bookmarkItem</upnp:class>
    <upnp:deviceUDN serviceType="AVTransport:1"
      serviceId="AVTransport">
      uuid:2F5A2466-55EF-44af-953A-74DE96FF2B14
    </upnp:deviceUDN>
    <upnp:deviceUDN serviceType="RenderingControl:1"
      serviceId="RenderingControl">
      uuid:EF0DB408-3018-4e13-831A-8349CA543538
    </upnp:deviceUDN>
    <upnp:bookmarkedObjectID>1230131</upnp:bookmarkedObjectID>
    <dc:date>2003-03-21T15:21:22</dc:date>
    <upnp:stateVariableCollection serviceName="AVTransport">
```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="RelativeTimePosition">
    00:22:01
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>
```

<!-- End of *stateVariableValuePairs XML Document* -->

```
</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="pre-mix">
```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="Brightness">
    50
  </stateVariable>
  <stateVariable variableName="Sharpness">
    33
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>
```

<!-- End of *stateVariableValuePairs XML Document* -->

```
</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="post-mix">
```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
```

```

http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="Brightness">
    70
  </stateVariable>
  <stateVariable variableName="Sharpness">
    21
  </stateVariable>
  <!-- More state variable value pairs can
  be inserted here -->
  </stateVariableValuePairs>

```

<!-- End of stateVariableValuePairs XML Document -->

```

  </upnp:stateVariableCollection>
</item>
</DIDL-Lite>

```

2.6.11.2 Creating and Destroying Bookmarks

The control point can use the [GetFeatureList\(\)](#) action to determine whether the *BOOKMARK* feature is supported by the ContentDirectory service and retrieve the object IDs of all the bookmark root containers within the ContentDirectory service. After the control point has gathered state information from the relevant AVTransport and RenderingControl services through the respective [GetStateVariables\(\)](#) actions, the control point can then decide where to create the bookmark. It can then proceed and create the new bookmark within one of the exposed bookmark subtrees. Alternatively, the control point may decide to create the bookmark outside the bookmark subtrees. Bookmark items can be created anywhere in the ContentDirectory data structure. However, the ContentDirectory MUST create a reference item to that bookmark within one of the exposed bookmark subtrees. The location of that reference item within the bookmark subtrees is vendor dependent.

If a bookmark container has its [@restricted](#) property set to “0” and its [upnp:createClass](#) set to “[object.container.bookmarkFolder](#)”, then only a bookmark container can be created. If a bookmark container has its [@restricted](#) property set to “0” and its [upnp:createClass](#) set to “[object.container.bookmarkItem](#)”, then only a bookmark item can be created. If a bookmark container has its [@restricted](#) property set to “0” and the [upnp:createClass](#) property is not specified, then both bookmark container and bookmark item can be created within that container.

The [Elements](#) input argument of the [CreateObject\(\)](#) action contains the title of the bookmark ([dc:title](#)), the UDN of the device that contains the AVTransport service, the UDN of the device that contains the RenderingControl service, the bookmark timestamp, and the respective state snapshots. The output of the [CreateObject\(\)](#) action contains the object ID of the newly created bookmark object in the [ObjectID](#) output argument and the *DIDL-Lite XML Document*, describing the created bookmark object in the [Result](#) output argument.

The following paragraph shows an example invocation of the [CreateObject\(\)](#) action. When a bookmark is created, the associated content item must be updated to contain the object ID of the newly created bookmark in its [upnp:bookmarkID](#) property. In this example, “BC_001” is used as the parent container’s object ID.

Request :

```

CreateObject("BC_001", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="" parentID="BC_001" restricted="0">
    <dc:title>Gone with the wind</dc:title>
    <upnp:class>object.item.bookmarkItem</upnp:class>
    <upnp:deviceUDN serviceType="AVTransport:1"
      serviceId="AVTransport">
      uuid:2F5A2466-55EF-44af-953A-74DE96FF2B14

```

```

</upnp:deviceUDN>
<upnp:deviceUDN serviceType="RenderingControl:1"
  serviceId="RenderingControl">
  uuid:EF0DB408-3018-4e13-831A-8349CA543538
</upnp:deviceUDN>
<upnp:bookmarkedObjectID>1230131</upnp:bookmarkedObjectID>
<upnp:stateVariableCollection serviceName="AVTransport">

```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="RelativeTimePosition">
    00:22:01
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs XML Document* -->

```

</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="pre-mix">

```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="Brightness">
    50
  </stateVariable>
  <stateVariable variableName="Sharpness">
    33
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs XML Document* -->

```

</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="post-mix">

```

<!--

The following *stateVariableValuePairs* XML Document needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="Brightness">
    70
  </stateVariable>
  <stateVariable variableName="Sharpness">
    21
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>
```

<!-- End of *stateVariableValuePairs* XML Document -->

```
</upnp:stateVariableCollection>
</item>
</DIDL-Lite>)
```

Response:

```
CreateObject("bookmark-763215", "
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="bookmark-763215" parentID="BC_001" restricted="0">
    <dc:title>Gone with the wind</dc:title>
    <upnp:class>object.item.bookmarkItem</upnp:class>
    <upnp:deviceUDN serviceType="AVTransport:1"
      serviceId="AVTransport">
      uuid:2F5A2466-55EF-44af-953A-74DE96FF2B14
    </upnp:deviceUDN>
    <upnp:deviceUDN serviceType="RenderingControl:1"
      serviceId="RenderingControl">
      uuid:EF0DB408-3018-4e13-831A-8349CA543538
    </upnp:deviceUDN>
    <upnp:bookmarkedObjectID>1230131</upnp:bookmarkedObjectID>
    <dc:date>2003-04-21T15:33:44</dc:date>
    <upnp:stateVariableCollection serviceName="AVTransport">
```

<!--

The following *stateVariableValuePairs* XML Document needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
```

```

    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
        urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
        <stateVariable variableName="RelativeTimePosition">
            00:22:01
        </stateVariable>
        <!-- More state variable value pairs can
            be inserted here -->
        </stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs XML Document* -->

```

</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
    rcsInstanceType="pre-mix">

```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

    <?xml version="1.0" encoding="UTF-8"?>
    <stateVariableValuePairs
        xmlns="urn:schemas-upnp-org:av:avs"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="
            urn:schemas-upnp-org:av:avs
        http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
        <stateVariable variableName="Brightness">
            50
        </stateVariable>
        <stateVariable variableName="Sharpness">
            33
        </stateVariable>
        <!-- More state variable value pairs can
            be inserted here -->
        </stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs XML Document* -->

```

</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
    rcsInstanceType="post-mix">

```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

    <?xml version="1.0" encoding="UTF-8"?>
    <stateVariableValuePairs
        xmlns="urn:schemas-upnp-org:av:avs"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="
            urn:schemas-upnp-org:av:avs
        http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
        <stateVariable variableName="Brightness">
            70
        </stateVariable>
        <stateVariable variableName="Sharpness">
            21
        </stateVariable>

```

```
<!-- More state variable value pairs can
be inserted here -->
</stateVariableValuePairs>
```

```
<!-- End of stateVariableValuePairs XML Document -->
```

```
</upnp:stateVariableCollection>
</item>
</DIDL-Lite>")
```

The [DestroyObject\(\)](#) action destroys the bookmark object specified by the [ObjectID](#) argument. The ContentDirectory service is REQUIRED to maintain consistency; that is, when a bookmark is destroyed, the associated content item's [upnp:bookmarkID](#) property MUST be removed. Likewise, when a content item that contains bookmark references is destroyed, the corresponding bookmark items (and their reference items, if any) MUST also be destroyed.

The following is an example of a [DestroyObject\(\)](#) action invocation:

Request:

```
DestroyObject ("bookmark-763215")
```

Response:

```
DestroyObject ()
```

2.6.11.3 Browsing Bookmarks

The bookmark list is obtained by invoking the [Browse\(\)](#) action with the [BrowseFlag](#) argument set to "[BrowseDirectChildren](#)". The [GetFeatureList\(\)](#) action MAY be used to find the bookmark root containers in the ContentDirectory service.

The following is an example where "BC_001" is used as the parent container's object ID.

Request:

```
Browse ("BC_001", "BrowseDirectChildren", "*", 0, 2, "")
```

Response:

```
Browse ("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
      http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
      http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="bookmark-00001" parentID="BC_001" restricted="0">
    <dc:title>The Matrix</dc:title>
    <upnp:class>object.item.bookmarkItem</upnp:class>
    <upnp:deviceUDN serviceType="AVTransport:1"
      serviceId="AVTransport">
      uuid:2F5A2466-55EF-44af-953A-74DE96FF2B14
    </upnp:deviceUDN>
    <upnp:deviceUDN serviceType="RenderingControl:1"
      serviceId="RenderingControl">
      uuid:EF0DB408-3018-4e13-831A-8349CA543538
    </upnp:deviceUDN>
    <upnp:bookmarkedObjectID>1230131</upnp:bookmarkedObjectID>
    <dc:date>2003-04-21T15:33:44</dc:date>
    <upnp:stateVariableCollection serviceName="AVTransport">
```

```
<!--
```

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="RelativeTimePosition">
    01:01:21
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>
```

<!-- End of *stateVariableValuePairs XML Document* -->

```
</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="pre-mix">
```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="Brightness">
    40
  </stateVariable>
  <stateVariable variableName="Sharpness">
    27
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>
```

<!-- End of *stateVariableValuePairs XML Document* -->

```
</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="post-mix">
```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
```

```

http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
    <stateVariable variableName="Brightness">
        70
    </stateVariable>
    <stateVariable variableName="Sharpness">
        21
    </stateVariable>
    <!-- More state variable value pairs can
    be inserted here -->
    </stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs XML Document* -->

```

    </upnp:stateVariableCollection>
</item>
<item id="bookmark-00002" parentID="BC_001" restricted="0">
    <dc:title>The Matrix Reloaded</dc:title>
    <upnp:class>object.item.bookmarkItem</upnp:class>
    <upnp:deviceUDN serviceType="AVTransport:1"
    serviceId="AVTransport">
        uuid:858733A8-E64C-4a2b-A407-38518D96AA0E
    </upnp:deviceUDN>
    <upnp:deviceUDN serviceType="RenderingControl:1"
    serviceId="RenderingControl">
        uuid:65AD5B9D-557E-4ddb-8EDE-F5A4C5190E57
    </upnp:deviceUDN>
    <upnp:bookmarkedObjectID>1230131</upnp:bookmarkedObjectID>
    <dc:date>2003-04-18T15:33:44</dc:date>
    <upnp:stateVariableCollection serviceName="AVTransport">

```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

    <?xml version="1.0" encoding="UTF-8"?>
    <stateVariableValuePairs
    xmlns="urn:schemas-upnp-org:av:avs"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
        urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
        <stateVariable variableName="RelativeTimePosition">
            01:55:22
        </stateVariable>
        <!-- More state variable value pairs can
        be inserted here -->
        </stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs XML Document* -->

```

    </upnp:stateVariableCollection>
    <upnp:stateVariableCollection serviceName="RenderingControl"
    rcsInstanceType="pre-mix">

```

<!--

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

    <?xml version="1.0" encoding="UTF-8"?>
    <stateVariableValuePairs
    xmlns="urn:schemas-upnp-org:av:avs"

```

```

    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
        urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
        <stateVariable variableName="Brightness">
            30
        </stateVariable>
        <stateVariable variableName="Sharpness">
            23
        </stateVariable>
        <!-- More state variable value pairs can
            be inserted here -->
        </stateVariableValuePairs>
<!-- End of stateVariableValuePairs XML Document -->

</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
    rcsInstanceType="post-mix">

<!--
The following stateVariableValuePairs XML Document needs to be interpreted
as a simple string and therefore needs to be properly escaped
-->

```

```

    <?xml version="1.0" encoding="UTF-8"?>
    <stateVariableValuePairs
        xmlns="urn:schemas-upnp-org:av:avs"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="
            urn:schemas-upnp-org:av:avs
        http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
        <stateVariable variableName="Brightness">
            70
        </stateVariable>
        <stateVariable variableName="Sharpness">
            21
        </stateVariable>
        <!-- More state variable value pairs can
            be inserted here -->
        </stateVariableValuePairs>
<!-- End of stateVariableValuePairs XML Document -->

    </upnp:stateVariableCollection>
</item>
</DIDL-Lite> ", 2, 2, 10)

```

Utilizing filters will reduce the size of the response. Once the user has selected a certain bookmark, another [Browse\(\)](#) action can be invoked to obtain the rest of the bookmark information:

Request:

```

Browse("BC_001", "BrowseDirectChildren", "@id, @parentId, @restricted,
dc:title, upnp:class, dc:date", 0, 2, "")

```

Response:

```

Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
    xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

```

```
xsi:schemaLocation="
urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
urn:schemas-upnp-org:metadata-1-0/upnp/
http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
<item id="bookmark-00001" parentID="BC_001" restricted="0">
  <dc:title>The Matrix</dc:title>
  <upnp:class>object.item.bookmarkItem</upnp:class>
  <dc:date>2003-04-13T15:33:44</dc:date>
</item>
<item id="bookmark-00002" parentID="BC_001" restricted="0">
  <dc:title>The Matrix Reloaded</dc:title>
  <upnp:class>object.item.bookmarkItem</upnp:class>
  <dc:date>2003-04-22T15:33:44</dc:date>
</item>
</DIDL-Lite>", 2, 2, 10)
```

The following example shows how to browse the container metadata of a bookmark container:

Request:

```
Browse("BC_001", "BrowseMetaData", "*", 0, 0, "")
```

Response:

```
Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <container id="BC_001" parentID="0" restricted="0"
    neverPlayable="1">
    <dc:title>BookMark Container</dc:title>
    <upnp:class>object.container.bookmarkFolder</upnp:class>
  </container>
</DIDL-Lite>, 1, 1, 20)
```

To obtain a particular bookmark, the bookmark id MUST be provided in the ObjectID argument and the BrowseFlag argument must be set to "BrowseMetaData". The following is an example:

Request:

```
Browse("bookmark-00001", "BrowseMetaData", "*", 0, 0, "")
```

Response:

```
Browse("
<?xml version="1.0" encoding="UTF-8"?>
<DIDL-Lite
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
  xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/
    http://www.upnp.org/schemas/av/didl-lite-v2-20060531.xsd
    urn:schemas-upnp-org:metadata-1-0/upnp/
    http://www.upnp.org/schemas/av/upnp-v2-20060531.xsd">
  <item id="bookmark-00001" parentID="BC_001" restricted="0">
    <dc:title>The Matrix</dc:title>
```

```

<upnp:class>object.item.bookmarkItem</upnp:class>
<upnp:deviceUDN serviceType="AVTransport:1"
  serviceId="AVTransport">
  uuid:2F5A2466-55EF-44af-953A-74DE96FF2B14
</upnp:deviceUDN>
<upnp:deviceUDN serviceType="RenderingControl:1"
  serviceId="RenderingControl">
  uuid:EF0DB408-3018-4e13-831A-8349CA543538
</upnp:deviceUDN>
<upnp:bookmarkedObjectID>1230131</upnp:bookmarkedObjectID>
<dc:date>2003-04-17T15:33:44</dc:date>
<upnp:stateVariableCollection serviceName="AVTransport">

```

<!--

The following *stateVariableValuePairs* XML Document needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="RelativeTimePosition">
    01:01:21
  </stateVariable>
  <!-- More state variable value pairs can
  be inserted here -->
</stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs* XML Document -->

```

</upnp:stateVariableCollection>
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="pre-mix">

```

<!--

The following *stateVariableValuePairs* XML Document needs to be interpreted as a simple string and therefore needs to be properly escaped

-->

```

<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="Brightness">
    40
  </stateVariable>
  <stateVariable variableName="Sharpness">
    27
  </stateVariable>
  <!-- More state variable value pairs can
  be inserted here -->
</stateVariableValuePairs>

```

<!-- End of *stateVariableValuePairs* XML Document -->

```

</upnp:stateVariableCollection>

```

```
<upnp:stateVariableCollection serviceName="RenderingControl"
  rcsInstanceType="post-mix">
```

```
<!--
```

The following *stateVariableValuePairs XML Document* needs to be interpreted as a simple string and therefore needs to be properly escaped

```
-->
```

```
<?xml version="1.0" encoding="UTF-8"?>
<stateVariableValuePairs
  xmlns="urn:schemas-upnp-org:av:avs"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    urn:schemas-upnp-org:av:avs
    http://www.upnp.org/schemas/av/avs-v1-20060531.xsd">
  <stateVariable variableName="Brightness">
    70
  </stateVariable>
  <stateVariable variableName="Sharpness">
    21
  </stateVariable>
  <!-- More state variable value pairs can
    be inserted here -->
</stateVariableValuePairs>
```

```
<!-- End of stateVariableValuePairs XML Document -->
```

```
</upnp:stateVariableCollection>
</item>
</DIDL-Lite>, 1, 1, 30)
```

2.6.12 Vendor Metadata Extensions

Vendors MAY extend DIDL-Lite metadata by placing blocks of vendor-specific metadata into `<desc>` blocks. In DIDL-Lite, a `<desc>` element identifies a *descriptor*. The REQUIRED `nameSpace` attribute identifies the namespace of the contained metadata. `<desc>` elements MAY appear as child elements of `<DIDL-Lite>` root elements, `<container>`, and `<item>` elements. The contents of each `<desc>` MUST be associated with only one namespace.

A *descriptor* is employed to associate blocks of other XML-based metadata with a given ContentDirectory service object. Examples of other XML-based metadata include DIG35, MPEG7, RDF, XrML, etc. *descriptor* blocks could also be employed to contain vendor-specific content ratings information, digitally signed rights descriptions, etc.

Allowing the `<desc>` elements to contain only elements from the namespace defined by the `nameSpace` attribute allows control point vendors to selectively deploy support for a given namespace using parser *plug-in* techniques. `<desc>` elements that have an unknown namespace specified in their `nameSpace` attribute should be ignored by the control point.

3 XML Service Description

```

<?xml version="1.0"?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
  <specVersion>
    <major>1</major>
    <minor>0</minor>
  </specVersion>
  <actionList>
    <action>
      <name>GetSearchCapabilities</name>
      <argumentList>
        <argument>
          <name>SearchCaps</name>
          <direction>out</direction>
          <relatedStateVariable>
            SearchCapabilities
          </relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>GetSortCapabilities</name>
      <argumentList>
        <argument>
          <name>SortCaps</name>
          <direction>out</direction>
          <relatedStateVariable>
            SortCapabilities
          </relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>GetSortExtensionCapabilities</name>
      <argumentList>
        <argument>
          <name>SortExtensionCaps</name>
          <direction>out</direction>
          <relatedStateVariable>
            SortExtensionCapabilities
          </relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>GetFeatureList</name>
      <argumentList>
        <argument>
          <name>FeatureList</name>
          <direction>out</direction>
          <relatedStateVariable>
            FeatureList
          </relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
      <name>GetSystemUpdateID</name>
      <argumentList>
        <argument>
          <name>Id</name>

```

```

        <direction>out</direction>
        <relatedStateVariable>
            SystemUpdateID
        </relatedStateVariable>
    </argument>
</argumentList>
</action>
<action>
    <name>Browse</name>
    <argumentList>
        <argument>
            <name>ObjectID</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE ObjectID
            </relatedStateVariable>
        </argument>
        <argument>
            <name>BrowseFlag</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE BrowseFlag
            </relatedStateVariable>
        </argument>
        <argument>
            <name>Filter</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE Filter
            </relatedStateVariable>
        </argument>
        <argument>
            <name>StartingIndex</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE Index
            </relatedStateVariable>
        </argument>
        <argument>
            <name>RequestedCount</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE Count
            </relatedStateVariable>
        </argument>
        <argument>
            <name>SortCriteria</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE SortCriteria
            </relatedStateVariable>
        </argument>
        <argument>
            <name>Result</name>
            <direction>out</direction>
            <relatedStateVariable>
                A ARG TYPE Result
            </relatedStateVariable>
        </argument>
        <argument>
            <name>NumberReturned</name>
            <direction>out</direction>
            <relatedStateVariable>

```

```

        A ARG TYPE Count
    </relatedStateVariable>
</argument>
<argument>
    <name>TotalMatches</name>
    <direction>out</direction>
    <relatedStateVariable>
        A ARG TYPE Count
    </relatedStateVariable>
</argument>
<argument>
    <name>UpdateID</name>
    <direction>out</direction>
    <relatedStateVariable>
        A ARG TYPE UpdateID
    </relatedStateVariable>
</argument>
</argumentList>
</action>
<action>
    <name>Search</name>
    <argumentList>
        <argument>
            <name>ContainerID</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE ObjectID
            </relatedStateVariable>
        </argument>
        <argument>
            <name>SearchCriteria</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE SearchCriteria
            </relatedStateVariable>
        </argument>
        <argument>
            <name>Filter</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE Filter
            </relatedStateVariable>
        </argument>
        <argument>
            <name>StartingIndex</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE Index
            </relatedStateVariable>
        </argument>
        <argument>
            <name>RequestedCount</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE Count
            </relatedStateVariable>
        </argument>
        <argument>
            <name>SortCriteria</name>
            <direction>in</direction>
            <relatedStateVariable>
                A ARG TYPE SortCriteria
            </relatedStateVariable>
        </argument>
    </argumentList>
</action>

```

```

</argument>
<argument>
  <name>Result</name>
  <direction>out</direction>
  <relatedStateVariable>
    A ARG TYPE Result
  </relatedStateVariable>
</argument>
<argument>
  <name>NumberReturned</name>
  <direction>out</direction>
  <relatedStateVariable>
    A ARG TYPE Count
  </relatedStateVariable>
</argument>
<argument>
  <name>TotalMatches</name>
  <direction>out</direction>
  <relatedStateVariable>
    A ARG TYPE Count
  </relatedStateVariable>
</argument>
<argument>
  <name>UpdateID</name>
  <direction>out</direction>
  <relatedStateVariable>
    A ARG TYPE UpdateID
  </relatedStateVariable>
</argument>
</argumentList>
</action>
<action>
  <name>CreateObject</name>
  <argumentList>
    <argument>
      <name>ContainerID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
    <argument>
      <name>Elements</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE Result
      </relatedStateVariable>
    </argument>
    <argument>
      <name>ObjectID</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
    <argument>
      <name>Result</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE Result
      </relatedStateVariable>
    </argument>
  </argumentList>

```

```

</action>
<action>
  <name>DestroyObject</name>
  <argumentList>
    <argument>
      <name>ObjectID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>UpdateObject</name>
  <argumentList>
    <argument>
      <name>ObjectID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
    <argument>
      <name>CurrentTagValue</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE TagValueList
      </relatedStateVariable>
    </argument>
    <argument>
      <name>NewTagValue</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE TagValueList
      </relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>MoveObject</name>
  <argumentList>
    <argument>
      <name>ObjectID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
    <argument>
      <name>NewParentID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
    <argument>
      <name>NewObjectID</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
  </argumentList>

```

```

    </argumentList>
</action>
<action>
  <name>ImportResource</name>
  <argumentList>
    <argument>
      <name>SourceURI</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE URI
      </relatedStateVariable>
    </argument>
    <argument>
      <name>DestinationURI</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE URI
      </relatedStateVariable>
    </argument>
    <argument>
      <name>TransferID</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE TransferID
      </relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>ExportResource</name>
  <argumentList>
    <argument>
      <name>SourceURI</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE URI
      </relatedStateVariable>
    </argument>
    <argument>
      <name>DestinationURI</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE URI
      </relatedStateVariable>
    </argument>
    <argument>
      <name>TransferID</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE TransferID
      </relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>StopTransferResource</name>
  <argumentList>
    <argument>
      <name>TransferID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE TransferID
      </relatedStateVariable>
    </argument>
  </argumentList>

```

```

    </argument>
  </argumentList>
</action>
<action>
  <name>DeleteResource</name>
  <argumentList>
    <argument>
      <name>ResourceURI</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE URI
      </relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>GetTransferProgress</name>
  <argumentList>
    <argument>
      <name>TransferID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE TransferID
      </relatedStateVariable>
    </argument>
    <argument>
      <name>TransferStatus</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE TransferStatus
      </relatedStateVariable>
    </argument>
    <argument>
      <name>TransferLength</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE TransferLength
      </relatedStateVariable>
    </argument>
    <argument>
      <name>TransferTotal</name>
      <direction>out</direction>
      <relatedStateVariable>
        A ARG TYPE TransferTotal
      </relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
  <name>CreateReference</name>
  <argumentList>
    <argument>
      <name>ContainerID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID
      </relatedStateVariable>
    </argument>
    <argument>
      <name>ObjectID</name>
      <direction>in</direction>
      <relatedStateVariable>
        A ARG TYPE ObjectID

```

```

        </relatedStateVariable>
    </argument>
    <argument>
        <name>NewID</name>
        <direction>out</direction>
        <relatedStateVariable>
            A ARG TYPE ObjectID
        </relatedStateVariable>
    </argument>
</argumentList>
</action>
Declarations for other actions added by UPnP vendor
(if any) go here
</actionList>
<serviceStateTable>
    <stateVariable sendEvents="no">
        <name>SearchCapabilities</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>SortCapabilities</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>SortExtensionCapabilities</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="yes">
        <name>SystemUpdateID</name>
        <dataType>ui4</dataType>
    </stateVariable>
    <stateVariable sendEvents="yes">
        <name>ContainerUpdateIDs</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="yes">
        <name>TransferIDs</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>FeatureList</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>A ARG TYPE ObjectID</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>A ARG TYPE Result</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>A ARG TYPE SearchCriteria</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>A ARG TYPE BrowseFlag</name>
        <dataType>string</dataType>
        <allowedValueList>
            <allowedValue>BrowseMetadata</allowedValue>
            <allowedValue>BrowseDirectChildren</allowedValue>
        </allowedValueList>
    </stateVariable>

```

```

<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_Filter</name>
  <dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_SortCriteria</name>
  <dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_Index</name>
  <dataType>ui4</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_Count</name>
  <dataType>ui4</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_UpdateID</name>
  <dataType>ui4</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_TransferID</name>
  <dataType>ui4</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_TransferStatus</name>
  <dataType>string</dataType>
  <allowedValueList>
    <allowedValue>COMPLETED</allowedValue>
    <allowedValue>ERROR</allowedValue>
    <allowedValue>IN_PROGRESS</allowedValue>
    <allowedValue>STOPPED</allowedValue>
  </allowedValueList>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_TransferLength</name>
  <dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_TransferTotal</name>
  <dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_TagValueList</name>
  <dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>A_ARG_TYPE_URI</name>
  <dataType>uri</dataType>
</stateVariable>
  Declarations for other state variables added by
  UPnP vendor (if any) go here
</serviceStateTable>
</scpd>

```

4 Test

No semantic tests have been specified for this service.

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008

Annex A (normative)

Schemas

This appendix describes the XML schemas for the DIDL-Lite element set. The UPnP, Dublin Core and XML namespaces are imported into the DIDL-Lite schema.

A.1 DIDL-Lite

DIDL-Lite is derived from a subset of DIDL, the *Digital Item Declaration Language*, recently developed within ISO/MPEG21 [DIDL].

The referenced DIDL-Lite schema [DIDL-LITE-XSD] may be downloaded from the UPnP Forum website and saved into a local file for use in a validating parser or instance document editing tool.

It is anticipated that few if any, UPnP A/V control points or ContentDirectory services will employ schema-based validation in the implementation of A/V functionality. The schema serves as a reference for the format of *DIDL-Lite XML Documents* and *DIDL-Lite XML Fragments*. Any discrepancies between this specification and the schema MUST be resolved in favor of the specification.

The schema however, may have a use in testing and certifying the UPnP A/V standard compliance of UPnP A/V control points and UPnP A/V ContentDirectory services (see Section 4, “Test.”)

The DIDL-Lite schema has been constructed using the May 2, 2001 W3C XML Schema Recommendation.

A.2 UPnP Elements

The referenced schema [UPNP-XSD] defines the *upnp* properties that are implemented as XML elements and attributes and used in DIDL-Lite. The schema may be downloaded from the UPnP Forum website and saved into a local file for use in a validating parser or instance document-editing tool.

A.3 Dublin Core Subset Elements

The referenced schema [DC-XSD] defines the *dc* namespace tags that are employed as descriptors under DIDL-Lite. They represent a subset of Dublin Core elements.

A.4 FeatureList State Variable Schema

The external XML schema [AVS-XSD] describes the format of the *FeatureList* state variable, which is used to indicate supported *CDS features* defined in Annex E, “

(normative)

CDS features”.

Annex B (normative)

AV Working Committee Properties

The tables and sections below list all properties of ContentDirectory service objects as defined by the AV Working Committee.

ContentDirectory service object descriptions are serialized into *DIDL-Lite XML Documents* in response to [Browse\(\)](#) and [Search\(\)](#) requests. *DIDL-Lite XML Documents* are formatted according to the DIDL-Lite schema in [DIDL-LITE-XSD]. The DIDL-Lite schema includes elements from the upnp schema [UPNP-XSD] and a subset of the Dublin Core schema [DC-XSD].

The tables and sections below describe each object property that can appear in serialized form in a *DIDL-Lite XML Document*, as well as the XML data type [XML SCHEMA-2] from which each property is derived. Properties that are directly based on XML datatypes are listed with the xsd: prefix.

Note: The NS column in the tables contains the namespace prefix of the namespace to which the property name belongs. The M-Val column indicates whether the property is multi-valued (M-Val = [YES](#)) or single-valued (M-Val = [NO](#)). See Section 2.2.2.

B.1 Base Properties

Table B-1: Base Properties Overview

Property Name	NS	Data Type	M-Val	Reference
@id	DIDL-Lite	xsd:string	NO	Appendix B.1.1
@parentID	DIDL-Lite	xsd:string	NO	Appendix B.1.2
@refID	DIDL-Lite	xsd:string	NO	Appendix B.1.3
@restricted	DIDL-Lite	xsd:boolean	NO	Appendix B.1.4
@searchable	DIDL-Lite	xsd:boolean	NO	Appendix B.1.5
@childCount	DIDL-Lite	xsd:unsignedInt	NO	Appendix B.1.6
dc:title	dc	xsd:string	NO	Appendix B.1.7
dc:creator	dc	xsd:string	NO	Appendix B.1.8
res	DIDL-Lite	xsd:anyURI	YES	Appendix B.1.9
upnp:class	upnp	xsd:string	NO	Appendix B.1.10
upnp:class@name	upnp	xsd:string	NO	Appendix B.1.10.1
upnp:searchClass	upnp	xsd:string	YES	Appendix B.1.11
upnp:searchClass@name	upnp	xsd:string	NO	Appendix B.1.11.1
upnp:searchClass@includeDerived	upnp	xsd:boolean	NO	Appendix B.1.11.2
upnp:createClass	upnp	xsd:string	YES	Appendix B.1.12
upnp:createClass@name	upnp	xsd:string	NO	Appendix B.1.12.1
upnp:createClass@includeDerived	upnp	xsd:boolean	NO	Appendix B.1.12.2
upnp:writeStatus	upnp	xsd:string	NO	Appendix B.1.13

B.1.1 @id

Namespace: DIDL-Lite

Property Data Type: xsd:string

Multi-Valued: NO

Description: The @id property is a REQUIRED property of an item or container object and contains an identifier for the object. The value of each object @id property MUST be unique with respect to the ContentDirectory service. It is highly RECOMMENDED that an object's @id property NOT change during the lifetime of the object. However, the @id property of an object MAY change, if absolutely necessary, for example, on reboot or when the object is moved by the MoveObject() action.

Default Value: N/A – The property is REQUIRED.

B.1.2 @parentID

Namespace: DIDL-Lite

Property Data Type: xsd:string

Multi-Valued: NO

Description: @parentID is a REQUIRED property of an item or container object. The @parentID property MUST be set and always remain equal to the @id property of the object's parent. The @parentID of the ContentDirectory service root container MUST be set to the reserved value of -1. The @parentID property of any other ContentDirectory service object MUST NOT take this value.

Default Value: N/A – The property is REQUIRED.

B.1.3 @refID

Namespace: DIDL-Lite

Property Data Type: xsd:string

Multi-Valued: NO

Description: The @refID property is only applicable to item objects. The presence of this property indicates that the item is actually referencing another existing item (*reference item*). The @refID property MUST be set and always remain equal to the @id property of the item that is referenced.

Default Value: None.

B.1.4 @restricted

Namespace: DIDL-Lite

Property Data Type: xsd:boolean

Multi-Valued: NO

Description: The REQUIRED @restricted property indicates whether the object is modifiable. If set to "1", the ability to modify a given object is confined to the ContentDirectory service. Control point metadata write access is disabled. If set to "0", a control point can modify the object's metadata and/or modify the object's children.

Default Value: N/A – The property is REQUIRED.

B.1.5 @searchable

Namespace: DIDL-Lite

Property Data Type: xsd:boolean

Multi-Valued: NO

Description: The @searchable property is only applicable to container objects. When "1" (true), the ability to perform a Search() action under a container is enabled, otherwise a Search() action under that container will return no results, even when child containers have their @searchable property set to "1".

Default Value: "0".

B.1.6 @childCount

Namespace: DIDL-Lite

Property Data Type: xsd:unsignedInt

Multi-Valued: NO

Description: The @childCount property is only applicable to container objects. It reflects the number of direct children contained in the container object.

Default Value: None.

B.1.7 dc:title

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: NO

Description: The dc:title property is a REQUIRED property and indicates a friendly name for the object. See <http://dublincore.org/documents/dces>.

Default Value: N/A – The property is REQUIRED.

B.1.8 dc:creator

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: NO

Description: The dc:creator property indicates an entity that owns the content or is primarily responsible for creating the content. Examples include a person, an organization or a service. Typically, the name of the creator should be used to indicate the entity. See <http://dublincore.org/documents/dces>.

Default Value: None.

B.1.9 res

Namespace: DIDL-Lite

Property Data Type: xsd:anyURI

Multi-Valued: YES

Description: The res property indicates a resource, typically a media file, associated with the object. If the value of the res property is not present, then the content has not yet been fully imported by the ContentDirectory service and is not yet accessible for playback purposes. Values MUST be properly escaped URIs as described in [RFC 2396].

Default Value: None.

B.1.10 upnp:class

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:class property is a REQUIRED property and it indicates the class of the object.

Default Value: N/A – The property is REQUIRED.

IECNORM.COM : Click to view the full PDF of ISO/IEC 29341-4-12:2008

B.1.10.1 allowedValueList for the upnp:class Property**Table B-2: allowedValueList for the upnp:class Property**

Value	R/O	Description
<u>“object.item”</u>	<u>Q</u>	See Appendix C.2.1
<u>“object.item.imageItem”</u>	<u>Q</u>	See Appendix C.2.1.1
<u>“object.item.imageItem.photo”</u>	<u>Q</u>	See Appendix C.2.1.1.1
<u>“object.item.audioItem”</u>	<u>Q</u>	See Appendix C.2.1.2
<u>“object.item.audioItem.musicTrack”</u>	<u>Q</u>	See Appendix C.2.1.2.1
<u>“object.item.audioItem.audioBroadcast”</u>	<u>Q</u>	See Appendix C.2.1.2.2
<u>“object.item.audioItem.audioBook”</u>	<u>Q</u>	See Appendix C.2.1.2.3
<u>“object.item.videoItem”</u>	<u>Q</u>	See Appendix C.2.1.3
<u>“object.item.videoItem.movie”</u>	<u>Q</u>	See Appendix C.2.1.3.1
<u>“object.item.videoItem.videoBroadcast”</u>	<u>Q</u>	See Appendix C.2.1.3.2
<u>“object.item.videoItem.musicVideoClip”</u>	<u>Q</u>	See Appendix C.2.1.3.3
<u>“object.item.playlistItem”</u>	<u>Q</u>	See Appendix C.2.1.4
<u>“object.item.textItem”</u>	<u>Q</u>	See Appendix C.2.1.5
<u>“object.item.bookmarkItem”</u>	<u>Q</u>	See Appendix C.2.1.6
<u>“object.item.epgItem”</u>	<u>Q</u>	See Appendix C.2.1.7
<u>“object.item.epgItem.audioProgram”</u>	<u>Q</u>	See Appendix C.2.1.7.1
<u>“object.item.epgItem.videoProgram”</u>	<u>Q</u>	See Appendix C.2.1.7.2
<u>“object.container.person”</u>	<u>Q</u>	See Appendix C.2.2.1
<u>“object.container.person.musicArtist”</u>	<u>Q</u>	See Appendix C.2.2.1.1
<u>“object.container.playlistContainer”</u>	<u>Q</u>	See Appendix C.2.2.2
<u>“object.container.album”</u>	<u>Q</u>	See Appendix C.2.2.3
<u>“object.container.album.musicAlbum”</u>	<u>Q</u>	See Appendix C.2.2.3.1
<u>“object.container.album.photoAlbum”</u>	<u>Q</u>	See Appendix C.2.2.3.2
<u>“object.container.genre”</u>	<u>Q</u>	See Appendix C.2.2.4
<u>“object.container.genre.musicGenre”</u>	<u>Q</u>	See Appendix C.2.2.4.1
<u>“object.container.genre.movieGenre”</u>	<u>Q</u>	See Appendix C.2.2.4.2
<u>“object.container.channelGroup”</u>	<u>Q</u>	See Appendix C.2.2.5
<u>“object.container.channelGroup.audioChannelGroup”</u>	<u>Q</u>	See Appendix C.2.2.5.1
<u>“object.container.channelGroup.videoChannelGroup”</u>	<u>Q</u>	See Appendix C.2.2.5.2
<u>“object.container.epgContainer”</u>	<u>Q</u>	See Appendix C.2.2.6
<u>“object.container.storageSystem”</u>	<u>Q</u>	See Appendix C.2.2.7
<u>“object.container.storageVolume”</u>	<u>Q</u>	See Appendix C.2.2.8
<u>“object.container.storageFolder”</u>	<u>Q</u>	See Appendix C.2.2.9
<u>“object.container.bookmarkFolder”</u>	<u>Q</u>	See Appendix C.2.2.10
<i>Vendor-defined</i>	<u>X</u>	

B.1.10.2 upnp:class@name

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:class@name property indicates a friendly name for the class of the object. This SHOULD NOT be used for class-based searches as it is not guaranteed to be unique or consistent across content items of the same class.

Default Value: None.

B.1.11 upnp:searchClass

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YES

Description: The upnp:searchClass property is only applicable to container objects. It contains a class for which the container object can be searched.

If @searchable = "I", then

- If no upnp:searchClass properties are specified, then the Search() action can return any match.
- If upnp:searchClass properties are specified, then the Search() action MUST only return matches from the classes specified in the upnp:searchClass properties.
- upnp:searchClass is OPTIONAL.
- upnp:searchClass is always determined by the ContentDirectory service.
- upnp:searchClass semantics are per container, there is no parent-child relationship, they only apply to searches started from that container.

else

- The container and its subtrees are not searchable.
- The values of the upnp:searchClass properties are meaningless and therefore the upnp:searchClass properties SHOULD NOT be included.

Default Value: If @searchable = "I", then all classes can be searched.

B.1.11.1 upnp:searchClass@name

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:searchClass property indicates a friendly name for the class.

Default Value: None.

B.1.11.2 upnp:searchClass@includeDerived

Namespace: upnp

Property Data Type: xsd:boolean

Multi-Valued: NO

Description: The upnp:searchClass@includeDerived property is a REQUIRED property of the associated upnp:searchClass property and indicates whether the class specified MUST also include derived classes. When set to "1", derived classes must be included. When set to "0", derived classes must be excluded.

Default Value: N/A – The property is REQUIRED when the upnp:searchClass property is present.

B.1.12 upnp:createClass

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YES

Description: The upnp:createClass property is only applicable to container objects. It contains a class that can be created within the container object.

If @restricted = "0", then

- If no upnp:createClass properties are specified, then CreateObject() MAY create any class of object under the container.
- If upnp:createClass properties are specified, then CreateObject() MUST only create classes of objects specified in the upnp:createClass properties.

- upnp:createClass is OPTIONAL.
- upnp:createClass semantics are per container, there is no parent-child relationship, they only apply to CreateObject() actions in that container.

else

- CreateObject() MUST fail since the container can not be modified.

Default Value: If @restricted = “0”, then any class of object MAY be created under the container.

B.1.12.1 upnp:createClass@name

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:createClass property indicates a friendly name for the class.

Default Value: None.

B.1.12.2 upnp:createClass@includeDerived

Namespace: upnp

Property Data Type: xsd:boolean

Multi-Valued: NO

Description: The upnp:createClass@includeDerived property is a REQUIRED property of the associated upnp:createClass property and indicates that the class specified also includes derived classes. When set to “1”, derived classes must be included. When set to “0”, derived classes must be excluded.

Default Value: N/A – The property is REQUIRED when the upnp:createClass property is present.

B.1.13 upnp:writeStatus

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:writeStatus property controls the modifiability of the resources of a given object. The ability for a control point to change the value of the upnp:writeStatus property is implementation dependent.

Default Value: “UNKNOWN”.

B.1.13.1 allowedValueList for the upnp:writeStatus Property

Table B-3: allowedValueList for the upnp:writeStatus Property

Value	R/O	Description
“ <u>WRITABLE</u> ”	<u>0</u>	The object’s resource(s) MAY be deleted and/or modified.
“ <u>PROTECTED</u> ”	<u>0</u>	The object’s resource(s) MAY NOT be deleted and/or modified.
“ <u>NOT WRITABLE</u> ”	<u>0</u>	The object’s resource(s) MAY NOT be modified.
“ <u>UNKNOWN</u> ”	<u>0</u>	The object’s resource(s) write status is unknown.
“ <u>MIXED</u> ”	<u>0</u>	Some of the object’s resource(s) have a different write status.

B.2 Resource Encoding Characteristics Properties

Table B-4: Resource Encoding Characteristics Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>res</u>	DIDL-Lite	xsd:anyURI	<u>NO</u>	Appendix B.2.1
<u>res@protocolInfo</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.1
<u>res@importUri</u>	DIDL-Lite	xsd:anyURI	<u>NO</u>	Appendix B.2.1.2
<u>res@size</u>	DIDL-Lite	xsd:unsignedLong	<u>NO</u>	Appendix B.2.1.3
<u>res@duration</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.4
<u>res@protection</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.5
<u>res@bitrate</u>	DIDL-Lite	xsd:unsignedInt	<u>NO</u>	Appendix B.2.1.6
<u>res@bitsPerSample</u>	DIDL-Lite	xsd:unsignedInt	<u>NO</u>	Appendix B.2.1.7
<u>res@sampleFrequency</u>	DIDL-Lite	xsd:unsignedInt	<u>NO</u>	Appendix B.2.1.8
<u>res@nrAudioChannels</u>	DIDL-Lite	xsd:unsignedInt	<u>NO</u>	Appendix B.2.1.9
<u>res@resolution</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.10
<u>res@colorDepth</u>	DIDL-Lite	xsd:unsignedInt	<u>NO</u>	Appendix B.2.1.11
<u>res@tspec</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.12
<u>res@allowedUse</u>	DIDL-Lite	CSV (xsd:string)	<u>NO</u>	Appendix B.2.1.13
<u>res@validityStart</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.14
<u>res@validityEnd</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.15
<u>res@remainingTime</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.16
<u>res@usageInfo</u>	DIDL-Lite	xsd:string	<u>NO</u>	Appendix B.2.1.17
<u>res@rightsInfoURI</u>	DIDL-Lite	xsd:anyURI	<u>NO</u>	Appendix B.2.1.18
<u>res@contentInfoURI</u>	DIDL-Lite	xsd:anyURI	<u>NO</u>	Appendix B.2.1.19
<u>res@recordQuality</u>	DIDL-Lite	CSV (xsd:string)	<u>NO</u>	Appendix B.2.1.20

B.2.1 [res](#)

See Appendix B.1.9, “[res](#)”.

B.2.1.1 [res@protocolInfo](#)

Namespace: DIDL-Lite

Property Data Type: xsd:string

Multi-Valued: [NO](#)

Description: This REQUIRED property identifies the protocol that MUST be used to transmit the resource (see also UPnP A/V Connection Manager Service template, Section 2.5.2).

Default Value: N/A – The property is REQUIRED when the [res](#) property is present.

B.2.1.2 **res@importUri****Namespace:** DIDL-Lite**Property Data Type:** xsd:anyURI**Multi-Valued:** NO

Description: The res@importUri property indicates the URI via which the resource can be imported to the ContentDirectory service via the ImportResource() action or HTTP POST. The res@importUri property identifies a *download portal* for the associated res property of a specific target object. It is used to create a local copy of the external content. After the transfer finishes successfully, the local content is then associated with the target object by setting the target object's res property value to a URI for that content, which MAY or MAY NOT be the same URI as the one specified in the res@importUri property, depending on the ContentDirectory service implementation.

Default Value: None.**B.2.1.3** **res@size****Namespace:** DIDL-Lite**Property Data Type:** xsd:unsignedLong**Multi-Valued:** NO

Description: The res@size property indicates the size in bytes of the resource. This property is a 64-bit unsigned integer.

Default Value: None.**B.2.1.4** **res@duration****Namespace:** DIDL-Lite**Property Data Type:** xsd:string**Multi-Valued:** NO

Description: The res@duration property indicates the time duration of the playback of the resource, at normal speed. The form of the duration string is:

H+ :MM :SS [. F+]

or

H+ :MM :SS [. F0 / F1]

where:

H+: one or more digits to indicate elapsed hours,

MM: exactly 2 digits to indicate minutes (00 to 59),

SS: exactly 2 digits to indicate seconds (00 to 59),

F+: any number of digits (including no digits) to indicate fractions of seconds,

F0/F1: a fraction, with F0 and F1 at least one digit long, and F0 < F1.

The string MAY be preceded by a “+” or “-” sign, and the decimal point itself MUST be omitted if there are no fractional second digits.

Default Value: None.**B.2.1.5** **res@protection****Namespace:** DIDL-Lite**Property Data Type:** xsd:string**Multi-Valued:** NO

Description: The res@protection property contains some identification of a protection system used for the resource.

Default Value: None.

B.2.1.6 **res@bitrate**

Namespace: DIDL-Lite

Property Data Type: xsd:unsignedInt

Multi-Valued: NO

Description: The res@bitrate property indicates the bitrate in **bytes/second** of the encoding of the resource.

Note that there exists an inconsistency with a res@bitrate property name and its value being expressed in bytes/sec.

In case the resource has been encoded using variable bitrate (VBR), it is RECOMMENDED that the res@bitrate value represents the average bitrate, calculated over the entire duration of the resource (total number of bytes divided by the total duration of the resource).

The res@bitrate value SHOULD NOT be taken as sufficient from a QoS or other perspective to prepare for the stream; The protocol used and the physical layer headers may increase the actual required bandwidth.

Default Value: None.

B.2.1.7 **res@bitsPerSample**

Namespace: DIDL-Lite

Property Data Type: xsd:unsignedInt

Multi-Valued: NO

Description: The res@bitsPerSample property indicates the number of bits used to represent one sample of the resource.

Default Value: None.

B.2.1.8 **res@sampleFrequency**

Namespace: DIDL-Lite

Property Data Type: xsd:unsignedInt

Multi-Valued: NO

Description: The res@sampleFrequency property indicates the sample frequency used to digitize the audio resource. Expressed in Hz.

Default Value: None.

B.2.1.9 **res@nrAudioChannels**

Namespace: DIDL-Lite

Property Data Type: xsd:unsignedInt

Multi-Valued: NO

Description: The res@nrAudioChannels property indicates the number of audio channels present in the audio resource, for example, 1 for mono, 2 for stereo, 6 for Dolby Surround.

Default Value: None.

B.2.1.10 **res@resolution**

Namespace: DIDL-Lite

Property Data Type: xsd:string

Multi-Valued: NO

Description: The res@resolution property indicates the XxY resolution, in pixels, of the resource (typically an imageItem or videoItem). The string pattern is of the form: “[0-9]+x[0-9]+” (one or more digits, followed by “x”, followed by one or more digits).

Default Value: None.

B.2.1.11 **res@colorDepth**

Namespace: DIDL-Lite

Property Data Type: xsd:unsignedInt

Multi-Valued: NO

Description: The res@colorDepth property indicates the number of bits per pixel used to represent the video or image resource.

Default Value: None.

B.2.1.12 res@tspec**Namespace:** DIDL-Lite**Property Data Type:** xsd:string**Multi-Valued:** NO

Description: The res@tspec property identifies the content QoS (quality of service) requirements. It has a maximum length of 256 characters. The details about this property, including its components and formatting constraints, are defined in the QoS Manager service definition document of the UPnP QoS protocol.

Default Value: None.

B.2.1.13 res@allowedUse**Namespace:** DIDL-Lite**Property Data Type:** CSV (xsd:string)**Multi-Valued:** NO

Description: The res@allowedUse property is composed of a comma-separated list of value pairs. Each value pair is composed of an enumerated string value, followed by a colon (“:”), followed by an integer. For example, “PLAY:5,COPY:1”.

In each pair, the first value corresponds to an allowed use for the resource referenced by the associated res property. RECOMMENDED enumerated values are: “PLAY”, “COPY”, “MOVE” and “UNKNOWN”. Vendors may extend this list. The “UNKNOWN” value is the default value when new resources are created. A value of “UNKNOWN” indicates that allowed uses for this resource may exist, but have not been reflected in the ContentDirectory service.

Any resource that has accompanying constraints on uses must expose a value for the res@allowedUse property. Any use of the resource that does not appear explicitly in the res@allowedUse property must be assumed to be disallowed. When the res@allowedUse property is not present, there are no use constraints on the resource.

The second quantity corresponds to the number of times the specified use is allowed to occur. A value of “-1” indicates that there is no limit on the number of times this use may occur.

It is recommended to update this value when the number of allowed uses changes. For example, a resource with the res@allowedUse property initially set to “COPY:1” should be updated to “COPY:0” after a copy has been successfully completed.

Default Value: “UNKNOWN”.

B.2.1.14 res@validityStart**Namespace:** DIDL-Lite**Property Data Type:** xsd:string**Multi-Valued:** NO

Description: The res@validityStart property defines the beginning date&time when the corresponding uses described in the res@allowedUse property become valid. The format of the res@validityStart property MUST comply with the date-time syntax as defined in Annex D.

The following example value designates May 30, 2004, 1:20pm, as a validity interval beginning value: “2004-05-30T13:20:00-05:00”.

When the res@validityStart property is not present, the beginning of the validity interval is assumed to have already started.

Default Value: The validity interval is assumed to have already started.

B.2.1.15 res@validityEnd**Namespace:** DIDL-Lite**Property Data Type:** xsd:string**Multi-Valued:** NO

Description: The res@validityEnd property defines the ending date&time when the corresponding uses described in the res@allowedUse property become invalid. The format of the res@validityEnd property MUST comply with the date-time syntax as defined in Annex D.

When the res@validityEnd property is not present, there correspondingly is no end to the validity interval.

Default Value: There is no end to the validity interval.

B.2.1.16 res@remainingTime

Namespace: DIDL-Lite

Property Data Type: xsd:string

Multi-Valued: NO

Description: The res@remainingTime property is used to indicate the amount of time remaining until the use specified in the res@allowedUse property is revoked. The remaining time is an aggregate amount of time that the resource may be used either continuously or in discrete intervals. When both res@remainingTime and res@validityEnd are specified, the use is revoked either when res@remainingTime reaches zero, or when the res@validityEnd time is reached, whichever occurs first. The format of the res@remainingTime property MUST comply with the duration syntax as defined in Annex D.

Example: “P08:03:10” indicates that the resource is available for an additional 8 hours, 3 minutes and 10 seconds.

Default Value: None.

B.2.1.17 res@usageInfo

Namespace: DIDL-Lite

Property Data Type: xsd:string

Multi-Valued: NO

Description: The res@usageInfo property contains a user-friendly string with additional information about the allowed use of the resource, as in the example: *“Playing of the movie is allowed in high-definition mode. One copy is allowed to be made, but only the standard definition version may be copied”*.

Default Value: None.

B.2.1.18 res@rightsInfoURI

Namespace: DIDL-Lite

Property Data Type: xsd:anyURI

Multi-Valued: NO

Description: The res@rightsInfoURI property references an html page and a web site associated with the rights vendor for the resource. The referenced page SHOULD assist the user interface in documenting the rights and the renewal of the allowed use of the resource.

Default Value: None.

B.2.1.19 res@contentInfoURI

Namespace: DIDL-Lite

Property Data Type: xsd:anyURI

Multi-Valued: NO

Description: Each res@contentInfoURI property contains a URI employed to assist the user interface in providing additional information to the user about the content referenced by the resource. The value of this property refers to an html page and a web site associated with the content vendor for the resource.

Default Value: None.

B.2.1.20 res@recordQuality

Namespace: DIDL-Lite

Property Data Type: CSV (xsd:string)

Multi-Valued: NO

Description: When the resource referenced by the res property was created by recording, the res@recordQuality property can be specified to indicate the quality level(s) used to make the recording. The res@recordQuality property is a CSV list of <type> “.” <recording quality> pairs. The type and quality in each pair are separated by a colon character (“.”). The type portion indicates what kind of value system is used in the recording quality portion. The recording quality portion is the actual recording quality value used. When there is more than one pair of colon-separated values in the list, all pairs MUST represent the same quality level in different type systems. For detailed descriptions of the type and quality values, refer to the properties srs:recordQuality@type and srs:recordQuality, respectively, as defined in the ScheduledRecording service specification [SRS].

Default Value: None.

B.3 Contributor-related Properties

Table B-5: Contributor-related Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:artist</u>	upnp	xsd:string	<u>YES</u>	Appendix B.3.1
<u>upnp:artist@role</u>	upnp	xsd:string	<u>NO</u>	Appendix B.3.1.1
<u>upnp:actor</u>	upnp	xsd:string	<u>YES</u>	Appendix B.3.2
<u>upnp:actor@role</u>	upnp	xsd:string	<u>NO</u>	Appendix B.3.2.1
<u>upnp:author</u>	upnp	xsd:string	<u>YES</u>	Appendix B.3.3
<u>upnp:author@role</u>	upnp	xsd:string	<u>NO</u>	Appendix B.3.3.1
<u>upnp:producer</u>	upnp	xsd:string	<u>YES</u>	Appendix B.3.4
<u>upnp:director</u>	upnp	xsd:string	<u>YES</u>	Appendix B.3.5
<u>dc:publisher</u>	dc	xsd:string	<u>YES</u>	Appendix B.3.6
<u>dc:contributor</u>	dc	xsd:string	<u>YES</u>	Appendix B.3.7

B.3.1 [upnp:artist](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: [YES](#)

Description: The [upnp:artist](#) property indicates the name of an artist.

Default Value: None.

B.3.1.1 [upnp:artist@role](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: [NO](#)

Description: The [upnp:artist@role](#) property indicates the role of the artist in the work.

Default Value: None.

B.3.2 [upnp:actor](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: [YES](#)

Description: The [upnp:actor](#) property indicates the name of an actor performing in (part of) the content.

Default Value: None.

B.3.2.1 [upnp:actor@role](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: [NO](#)

Description: The [upnp:actor@role](#) property indicates the role of the actor in the work.

Default Value: None.

B.3.3 [upnp:author](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: [YES](#)

Description: The [upnp:author](#) property indicates the name of an author contributing to the content (for example, the writer of a text book).

Default Value: None.

B.3.3.1 upnp:author@role

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NODescription: The upnp:author@role property indicates the role of the author in the work.

Default Value: None.

B.3.4 upnp:producer

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YESDescription: The upnp:producer property indicates the name of a producer of the content (for example, a movie or a CD).

Default Value: None.

B.3.5 upnp:director

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YESDescription: The upnp:director property indicates the name of a director of the content (for example, a movie).

Default Value: None.

B.3.6 dc:publisher

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: YESDescription: The dc:publisher property indicates the name of a publisher of the content. See <http://dublincore.org/documents/dces>.

Default Value: None.

B.3.7 dc:contributor

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: YESDescription: The dc:contributor property indicates the name of a contributor to the content item. It is RECOMMENDED that dc:contributor property includes the name of the primary content creator or owner (Dublin Core ‘creator’ property). See <http://dublincore.org/documents/dces>.

Default Value: None.

B.4 Affiliation-related Properties

Table B-6: Affiliation-related Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:genre</u>	upnp	xsd:string	<u>YES</u>	Appendix B.4.1
<u>upnp:genre@id</u>	upnp	xsd:string	<u>NO</u>	Appendix B.4.1.1
<u>upnp:genre@extended</u>	upnp	CSV (xsd:string)	<u>NO</u>	Appendix B.4.1.1
<u>upnp:album</u>	upnp	xsd:string	<u>YES</u>	Appendix B.4.2
<u>upnp:playlist</u>	upnp	xsd:string	<u>YES</u>	Appendix B.4.3

B.4.1 upnp:genre

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YES**Description:** The upnp:genre property indicates the genre to which an object belongs.**Default Value:** None.**B.4.1.1 upnp:genre@id**

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO**Description:** The upnp:genre@id property identifies the genre scheme which defines the set of names used in the upnp:genre and upnp:genre@extended property.The format of the upnp:genre@id is:

<ICANN registered domain> “_” <genre_scheme_id>.

Example: “epg.com_GenreSet1”

The upnp:genre@id property is REQUIRED if the upnp:genre@extended property is specified.**Default Value:** N/A – This property is REQUIRED when the upnp:genre@extended property is present.**B.4.1.2 upnp:genre@extended**

Namespace: upnp

Property Data Type: CSV (xsd:string)

Multi-Valued: NO**Description:** The upnp:genre@extended property MUST be a CSV list of genre names, which are individually displayable strings, representing increasingly precise (sub)genre names. The list MUST be ordered with the most general genre first. The first entry in the list MUST be equal to the value of the upnp:genre property.

Example: “Sports,Basketball,NBA”

Default Value: None.**B.4.2 upnp:album**

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YES**Description:** The upnp:album property indicates the title of the album to which the content item belongs.**Default Value:** None.**B.4.3 upnp:playlist**

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YES**Description:** The upnp:playlist property indicates the name of a playlist (the dc:title of a playlistItem) to which the content item belongs.**Default Value:** None.**B.5 Associated Resources Properties****Table B-7: Associated Resources Properties Overview**

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:albumArtURI</u>	upnp	xsd:anyURI	<u>YES</u>	Appendix B.5.1
<u>upnp:artistDiscographyURI</u>	upnp	xsd:anyURI	<u>NO</u>	Appendix B.5.2
<u>upnp:lyricsURI</u>	upnp	xsd:anyURI	<u>NO</u>	Appendix B.5.3
<u>dc:relation</u>	dc	xsd:string	<u>YES</u>	Appendix B.5.4

B.5.1 upnp:albumArtURI

Namespace: upnp

Property Data Type: xsd:anyURI

Multi-Valued: YES

Description: The upnp:albumArtURI property contains a reference to album art. The value MUST be a properly escaped URI as described in [RFC 2396].

Default Value: None.

B.5.2 upnp:artistDiscographyURI

Namespace: upnp

Property Data Type: xsd:anyURI

Multi-Valued: NO

Description: The upnp:artistDiscographyURI property contains a reference to the artist's discography. The value MUST be a properly escaped URI as described in [RFC 2396].

Default Value: None.

B.5.3 upnp:lyricsURI

Namespace: upnp

Property Data Type: xsd:anyURI

Multi-Valued: NO

Description: The upnp:lyricsURI property contains a reference to lyrics of the song or of the whole album. The value MUST be a properly escaped URI as described in [RFC 2396].

Default Value: None.

B.5.4 dc:relation

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: YES

Description: See <http://dublincore.org/documents/dces>. The value MUST be a properly escaped URI as described in [RFC 2396].

Default Value: None.

B.6 Storage-Related Properties

Table B-8: Storage-Related Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:storageTotal</u>	upnp	xsd:long	<u>NO</u>	Appendix B.6.1
<u>upnp:storageUsed</u>	upnp	xsd:long	<u>NO</u>	Appendix B.6.2
<u>upnp:storageFree</u>	upnp	xsd:long	<u>NO</u>	Appendix B.6.3
<u>upnp:storageMaxPartition</u>	upnp	xsd:long	<u>NO</u>	Appendix B.6.4
<u>upnp:storageMedium</u>	upnp	xsd:string	<u>NO</u>	Appendix B.6.5

B.6.1 upnp:storageTotal

Namespace: upnp

Property Data Type: xsd:long

Multi-Valued: NO

Description: The upnp:storageTotal property contains the total capacity, in bytes, of the storage represented by the container. Value -1 is reserved to indicate that the capacity is unknown.

Default Value: None.

B.6.2 upnp:storageUsed

Namespace: upnp

Property Data Type: xsd:long

Multi-Valued: NO

Description: The upnp:storageUsed property contains the combined space, in bytes, used by all the objects held in the storage represented by the container. Value -1 is reserved to indicate that the space is unknown.

Default Value: None.

B.6.3 upnp:storageFree

Namespace: upnp

Property Data Type: xsd:long

Multi-Valued: NO

Description: The upnp:storageFree property contains the total free capacity, in bytes, of the storage represented by the container. Value -1 is reserved to indicate that the capacity is unknown.

Default Value: None.

B.6.4 upnp:storageMaxPartition

Namespace: upnp

Property Data Type: xsd:long

Multi-Valued: NO

Description: The upnp:storageMaxPartition property contains the largest amount of space, in bytes, available for storing a single resource in the container. Value -1 is reserved to indicate that the amount of space is unknown.

Default Value: None.

B.6.5 upnp:storageMedium

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:storageMedium property indicates the type of storage medium used for the content. Potentially useful for user-interface purposes.

Default Value: “UNKNOWN”.

B.6.5.1 allowedValueList for the upnp:storageMedium Property

See Table 2-4, “allowedValueList for PlaybackStorageMedium and RecordStorageMedium” of the AVTransport:1 service specification.

B.7 General Description (mainly for UI purposes) Properties**Table B-9: General Description (mainly for UI purposes) Properties Overview**

Property Name	NS	Data Type	M-Val	Reference
<u>dc:description</u>	dc	xsd:string	<u>NO</u>	Appendix B.7.1
<u>upnp:longDescription</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.2
<u>upnp:icon</u>	upnp	xsd:anyURI	<u>NO</u>	Appendix B.7.3
<u>upnp:region</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.4
<u>dc:rights</u>	dc	xsd:string	<u>YES</u>	Appendix B.7.5
<u>dc:date</u>	dc	xsd:string	<u>NO</u>	Appendix B.7.6
<u>dc:language</u>	dc	xsd:string	<u>YES</u>	Appendix B.7.7
<u>upnp:playbackCount</u>	upnp	xsd:int	<u>NO</u>	Appendix B.7.8
<u>upnp:lastPlaybackTime</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.9
<u>upnp:lastPlaybackPosition</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.10
<u>upnp:recordedStartDateTime</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.11
<u>upnp:recordedDuration</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.12
<u>upnp:recordedDayOfWeek</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.13
<u>upnp:srsRecordScheduleID</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.14
<u>upnp:srsRecordTaskID</u>	upnp	xsd:string	<u>NO</u>	Appendix B.7.15
<u>upnp:recordable</u>	upnp	xsd:boolean	<u>NO</u>	Appendix B.7.16

B.7.1 dc:description

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: NO

Description: The dc:description property contains a brief description of the content item. See <http://dublincore.org/documents/dces>.

Default Value: None.

B.7.2 upnp:longDescription

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:longDescription property contains a few lines of description of the content item (longer than the dc:description property).

Default Value: None.

B.7.3 upnp:icon

Namespace: upnp

Property Data Type: xsd:anyURI

Multi-Valued: NO

Description: The upnp:icon property contains a URI to some icon that a control point can use in its UI to display the content, for example, a CNN logo for a Tuner channel. It is RECOMMENDED that the same format be used as is used for the icon element in the UPnP device description document schema (PNG). The value MUST be a properly escaped URI as described in [RFC 2396].

Default Value: None.

B.7.4 upnp:region

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:region property contains some identification of the region, associated with the source of the object, for example, “US”, “Latin America”, “Seattle”.

Default Value: None.

B.7.5 upnp:rights

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YES

Description: The upnp:rights property contains some descriptive information about the legal rights held in or over this resource. (<http://dublincore.org/documents/dces>)

Default Value: None.

B.7.6 dc:date

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: NO

Description: The dc:date property contains the primary date of the content. The format MUST be compliant to [ISO 8601] and SHOULD be compliant to [RFC 3339]. See <http://dublincore.org/documents/dces>.

Examples:

- 2004-05-14
- 2004-05-14T14:30:05
- 2004-05-14T14:30:05+09:00

Default Value: None.

B.7.7 dc:language

Namespace: dc

Property Data Type: xsd:string

Multi-Valued: YES

Description: The dc:language property indicates one of the languages used in the content as defined by RFC 3066, for example, “en-US”. See <http://dublincore.org/documents/dces>.

Default Value: None.

B.7.8 upnp:playbackCount

Namespace: upnp

Property Data Type: xsd:int

Multi-Valued: NO

Description: The upnp:playbackCount property contains the number of times the content has been played. The special value -1 means that the content has been played but the count is unknown. The criteria for determining whether the content has been played, is device dependent.

Default Value: None.

B.7.9 upnp:lastPlaybackTime

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:lastPlaybackTime property contains the date&time of the last playback.

The format of the upnp:lastPlaybackTime property MUST comply with the date-time syntax as defined in Annex D.

The criteria for determining when the content has been played last, is device dependent.

Default Value: None.

B.7.10 upnp:lastPlaybackPosition

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:lastPlaybackPosition property contains the time offset within the content where the last playback was suspended.

The format of the upnp:lastPlaybackPosition property MUST comply with the duration syntax as defined in Annex D.

The criteria for determining the time offset in the content where the playback of the content has been suspended, is device dependent.

Default Value: None.

B.7.11 upnp:recordedStartDateTime

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:recordedStartDateTime property contains the date&time when the recording started.

The format of the upnp:recordedStartDateTime property MUST comply with the date-time syntax as defined in Annex D.

Default Value: None.

B.7.12 upnp:recordedDuration

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:recordedDuration property contains the duration of the recorded content.

The format of the upnp:recordedDuration property MUST comply with the duration syntax as defined in Annex D.

Default Value: None.

B.7.13 upnp:recordedDayOfWeek

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:recordedDayOfWeek property contains the day of the week when the recording *started*.

Sorting for this property is based on the order in Table B-10. Ascending: first table entry first.

Default Value: None.

B.7.13.1 allowedValueList for the upnp:recordedDayOfWeek Property

Table B-10: allowedValueList for the upnp:recordedDayOfWeek Property

Value	R/O	Description
" <u>SUN</u> "	<u>R</u>	
" <u>MON</u> "	<u>R</u>	
" <u>TUE</u> "	<u>R</u>	
" <u>WED</u> "	<u>R</u>	
" <u>THU</u> "	<u>R</u>	
" <u>FRI</u> "	<u>R</u>	
" <u>SAT</u> "	<u>R</u>	

B.7.14 upnp:srsRecordScheduleID

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:srsRecordScheduleID property contains the value of the srs:@id property of the srs:recordSchedule object that was used to create this recorded content. Refer to the ScheduledRecording service specification [SRS] for details.

Default Value: None.

B.7.15 upnp:srsRecordTaskID

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:srsRecordTaskID property contains the value of the srs:@id property of the srs:recordTask object that was used to create this recorded content. Refer to the ScheduledRecording service specification [SRS] for details.

Default Value: None.

B.7.16 upnp:recordable

Namespace: upnp

Property Data Type: xsd:boolean

Multi-Valued: NO

Description: When set to "1", the content represented by this object can potentially be used for recording purposes. If the object is not self-contained (such as an object of class other than "object.item.epgItem"), other information may be needed to set up the recording. When set to "0", the content represented by this object is not accessible for recording due to various reasons, such as hardware limitations.

Default Value: "1".

B.8 Recorded Object-related Properties

Table B-11: Recorded Object-related Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:programTitle</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.1
<u>upnp:seriesTitle</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.2
<u>upnp:programID</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.3
<u>upnp:programID@type</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.3.1
<u>upnp:seriesID</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.4
<u>upnp:seriesID@type</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.4.1
<u>upnp:channelID</u>	upnp	xsd:string	<u>YES</u>	Appendix B.8.5
<u>upnp:channelID@type</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.5.1
<u>upnp:episodeCount</u>	upnp	xsd:unsignedInt	<u>NO</u>	Appendix B.8.6
<u>upnp:episodeNumber</u>	upnp	xsd:unsignedInt	<u>NO</u>	Appendix B.8.7
<u>upnp:programCode</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.8
<u>upnp:programCode@type</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.8.1
<u>upnp:rating</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.9
<u>upnp:rating@type</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.9.1
<u>upnp:episodeType</u>	upnp	xsd:string	<u>NO</u>	Appendix B.8.10

B.8.1 [upnp:programTitle](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:programTitle](#) property contains the name of the program. This is most likely obtained from a database that contains program-related information, such as an Electronic Program Guide.

Example: “Friends Series Finale”.

Note: To be precise, this is different from the [dc:title](#) property which indicates a friendly name for the ContentDirectory service *object*. However, in many cases, the [dc:title](#) property will be set to the same value as the [upnp:programTitle](#) property.

Default Value: None.

B.8.2 [upnp:seriesTitle](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:seriesTitle](#) property contains the name of the series. This is most likely obtained from a database that contains program-related information, such as an Electronic Program Guide.

Default Value: None.

B.8.3 upnp:programID

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:programID property contains the unique ID of a program.

When this content is created via a ScheduledRecording service, this is the value of the srs:matchedID property of the recordTask that generated this content. Otherwise, the upnp:programID property value is set by the ContentDirectory service based on some device dependent information, such as an EPG database.

The format and semantics are identical to those of the srs:matchedID property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

Default Value: None.

B.8.3.1 upnp:programID@type

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:programID@type property indicates the type of the ID that is contained in the upnp:programID property. The format and allowed values are identical to those of the srs:matchedID@type property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

The upnp:programID@type property is REQUIRED if the upnp:programID property is specified.

Default Value: N/A – The property is REQUIRED when the upnp:programID property is present.

B.8.4 upnp:seriesID

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:seriesID property contains the unique ID of a series.

When this content is created via a ScheduledRecording service, this is the value of the srs:matchedID property of the recordTask that generated this content. Otherwise, the upnp:seriesID property value is set by the ContentDirectory service based on some device dependent information, such as an EPG database.

The format and semantics are identical to those of the srs:matchedID property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

Default Value: None.

B.8.4.1 upnp:seriesID@type

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:seriesID@type property indicates the type of the ID that is contained in the upnp:seriesID property. The format and allowed values are identical to those of the srs:matchedID@type property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

The upnp:seriesID@type property is REQUIRED if the upnp:seriesID property is specified.

Default Value: N/A – The property is REQUIRED when the upnp:seriesID property is present.

B.8.5 upnp:channelID**Namespace:** upnp**Property Data Type:** xsd:string**Multi-Valued:** YES

Description: When this content is created via a ScheduledRecording service, the upnp:channelID property identifies the channel that was the source. Otherwise, the upnp:channelID value indicates the channel that is associated with the content item. For example, when present in an object that represents a tuner channel, it contains the ID of that channel.

The possible formats and the dependency on the upnp:channelID@type property are identical to the possible formats of the srs:scheduledChannelID and its dependency on the srs:scheduledChannelID@type property as described in the ScheduledRecording service [SRS].

The upnp:channelID property is multi-valued so that different formats can be used to identify a particular channel. For example, if both the analog channel number and the analog channel frequency are known for the same channel, they can be advertised through the following construct:

```
<upnp:channelID type="ANALOG">5</upnp:channelID>
<upnp:channelID type="FREQUENCY">79000000</upnp:channelID>
```

When multiple instances of the upnp:channelID property are included, they MUST refer to the same channel.

Default Value: None.

B.8.5.1 upnp:channelID@type**Namespace:** upnp**Property Data Type:** xsd:string**Multi-Valued:** NO

Description: The upnp:channelID@type property determines the format that is used for the upnp:channelID property as defined above.

The possible formats and allowed values of the upnp:channelID@type property are identical to the possible formats of the srs:scheduledChannelID@type property as described in the ScheduledRecording service specification [SRS].

The upnp:channelID@type property is REQUIRED if the upnp:channelID property is specified.

Default Value: N/A – The property is REQUIRED when the upnp:channelID property is present.

B.8.6 upnp:episodeCount**Namespace:** upnp**Property Data Type:** xsd:unsignedInt**Multi-Valued:** NO

Description: The upnp:episodeCount property contains the total number of episodes in the series to which this content belongs.

Default Value: None.

B.8.7 upnp:episodeNumber**Namespace:** upnp**Property Data Type:** xsd:unsignedInt**Multi-Valued:** NO

Description: The upnp:episodeCount property contains the episode number of this recorded content within the series to which this content belongs.

Default Value: None.

B.8.8 upnp:programCode

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:programCode property contains a unique program code.

When this content is created via a ScheduledRecording service, this is the value of the srs:taskProgramCode property of the recordTask that generated this content. Otherwise, the upnp:programCode property value is set by the ContentDirectory service based on some device dependent information, such as an EPG database.

The format and semantics are identical to those of the srs:taskProgramCode property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

Default Value: None.

B.8.8.1 upnp:programCode@type

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:programCode@type property indicates the type of the program guide service that defines the program code specified in the upnp:programCode property. The format and allowed values are identical to those of the srs:taskProgramCode@type property, defined in the ScheduledRecording service specification. See [SRS] for details.

The upnp:programCode@type property is REQUIRED if the upnp:programCode property is specified.

Default Value: N/A – The property is REQUIRED when the upnp:programCode property is present.

B.8.9 upnp:rating

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: YES

Description: The upnp:rating property contains the viewer rating value of the content of this item expressed in the rating system indicated by the upnp:rating@type property. The format and semantics of the upnp:rating property are identical to those of the srs:matchedRating property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

Default Value: None.

B.8.9.1 upnp:rating@type

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:rating@type property indicates the rating system used in the upnp:rating property. The format and allowed values of the upnp:rating@type property are identical to those of the srs:matchedRating@type property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

The upnp:rating@type property is highly RECOMMENDED if the upnp:rating property is specified.

Default Value: N/A.

B.8.10 upnp:episodeType

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:episodeType property value indicates the broadcast novelty (for example, “FIRST-RUN” or “REPEAT”) of this content item. The format and allowed values of the upnp:episodeType property are identical to those of the srs:matchedEpisodeType property, defined in the ScheduledRecording service specification. Refer to the ScheduledRecording service specification [SRS] for details.

Default Value: None.

B.9 User Channel and EPG Related Properties

Table B-12: User Channel and EPG Related Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:channelGroupName</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.1
<u>upnp:channelGroupName@id</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.1.1
<u>upnp:callSign</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.2
<u>upnp:networkAffiliation</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.3
<u>upnp:serviceProvider</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.4
<u>upnp:price</u>	upnp	xsd:float	<u>YES</u>	Appendix B.9.5
<u>upnp:price@currency</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.5.1
<u>upnp:payPerView</u>	upnp	xsd:boolean	<u>NO</u>	Appendix B.9.6
<u>upnp:epgProviderName</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.7
<u>upnp:dateTimeRange</u>	upnp	xsd:string	<u>NO</u>	Appendix B.9.8

B.9.1 [upnp:channelGroupName](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:channelGroupName](#) property contains the user friendly name of the channelGroup.

Examples: “Digital Terrestrial”, “DirecTV”

A channel group defines a group of channels. A device that has multiple tuners may provide multiple channel groups. Moreover, a physical tuner device may provide multiple channel groups (for example, a set-top-box that contains a single tuner but supports three different input connections: terrestrial, cable, and satellite).

In a channel group, channels may be identified in various ways. For example, [upnp:channelID](#), [upnp:channelName](#), or [upnp:channelNr](#) may be used for that purpose.

Default Value: None.

B.9.1.1 [upnp:channelGroupName@id](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:channelGroupName@id](#) property contains the ID of a channel group to differentiate it from other channel groups implemented in a ContentDirectory service.

The format of the [upnp:channelGroupName@id](#) property is as follows:

<ICANN registered domain> “_” <channel group id defined in the domain>

Example: “broadcast.com_DigitalSatellite”

The [upnp:channelGroupName@id](#) property is REQUIRED if the [upnp:channelGroupName](#) property is specified.

Default Value: N/A – The property is REQUIRED when the [upnp:channelGroupName](#) property is present.

B.9.2 [upnp:callSign](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:callSign](#) property contains the broadcast station call sign of the associated broadcast channel. This is typically used for live content or recorded content.

Example: “KGW”.

If the [upnp:callSign](#) property is supported and [upnp:class](#) = “[object.item.audioItem.audioBroadcast](#)” then the [upnp:radioCallSign](#) property MUST also be supported and MUST be set equal to the value of the [upnp:callSign](#) property.

Default Value: None.

B.9.3 upnp:networkAffiliation

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:networkAffiliation property contains the name of the broadcast network or distribution network associated with this content. This is typically used for live content or recorded content.

Examples: “NBC”, “CBS”, “BBC”.

Default Value: None.

B.9.4 upnp:serviceProvider

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:serviceProvider property contains the friendly name of the service provider of this content. This is typically used for live content or recorded content. Note that one service provider may provide multiple channel groups.

Examples: “CANAL+”, “Echostar”, “SkyLife”.

Default Value: None.

B.9.5 upnp:price

Namespace: upnp

Property Data Type: xsd:float

Multi-Valued: YES

Description: The upnp:price property contains the price for a broadcast, series, program, movie, etc.

Default Value: None.

B.9.5.1 upnp:price@currency

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:price@currency property indicates the unit of currency used for the upnp:price property.

The upnp:price@currency property is REQUIRED if the upnp:price property is specified.

Default Value: N/A – The property is REQUIRED when the upnp:price property is present.

B.9.6 upnp:payPerView

Namespace: upnp

Property Data Type: xsd:boolean

Multi-Valued: NO

Description: The upnp:payPerView property indicates whether the object represents pay-per-view content. When set to “1”, the object is a pay-per-view object. When set to “0”, the object is not a pay-per-view object.

Default Value: None.

B.9.7 upnp:epgProviderName

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:epgProviderName property indicates the name of the Electronic Program Guide service provider.

Default Value: None.

B.9.8 upnp:dateTimeRange

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The upnp:dateTimeRange property indicates that all EPG items found in this container’s subtree exist within this time range. The format of the upnp:dateTimeRange property MUST comply with the date-time-range syntax as defined in Annex D.

Default Value: None.

B.10 Radio Broadcast Properties

Table B-13: Radio Broadcast Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:radioCallSign</u>	upnp	xsd:string	<u>NO</u>	Appendix B.10.1
<u>upnp:radioStationID</u>	upnp	xsd:string	<u>NO</u>	Appendix B.10.2
<u>upnp:radioBand</u>	upnp	xsd:string	<u>NO</u>	Appendix B.10.3

B.10.1 [upnp:radioCallSign](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:radioCallSign](#) property contains a radio station call sign, for example, “KSJO”.

Default Value: None.

B.10.2 [upnp:radioStationID](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:radioStationID](#) property contains some identification, for example, “107.7”, broadcast frequency of the radio station.

Default Value: None.

B.10.3 [upnp:radioBand](#)

Namespace: upnp

Property Data Type: xsd:string

Multi-Valued: NO

Description: The [upnp:radioBand](#) property contains the radio station frequency band.

Default Value: None.

B.10.3.1 allowedValueList for the [upnp:radioBand](#) Property

Table B-14: allowedValueList for the upnp:radioBand Property

Value	R/O	Description
“ <u>AM</u> ”	<u>Q</u>	
“ <u>FM</u> ”	<u>Q</u>	
“ <u>Shortwave</u> ”	<u>Q</u>	
“ <u>Internet</u> ”	<u>Q</u>	
“ <u>Satellite</u> ”	<u>Q</u>	
<i>Vendor-defined</i>	<u>X</u>	

B.11 Video Broadcast Properties

Table B-15: Video Broadcast Properties Overview

Property Name	NS	Data Type	M-Val	Reference
<u>upnp:channelNr</u>	upnp	xsd:int	<u>NO</u>	Appendix B.11.1
<u>upnp:channelName</u>	upnp	xsd:string	<u>NO</u>	Appendix B.11.2
<u>upnp:scheduledStartTime</u>	upnp	xsd:string	<u>YES</u>	Appendix B.11.3
<u>upnp:scheduledEndTime</u>	upnp	xsd:string	<u>YES</u>	Appendix B.11.4