

INTERNATIONAL STANDARD

**Digital living network alliance (DLNA) home networked device interoperability
guidelines –
Part 1: Architecture and protocols**

With IEC Norm.com : Click to view the full PDF of IEC 62481-1:2007



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2007 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 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 It also gives information on projects, withdrawn and replaced publications.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

INTERNATIONAL STANDARD

**Digital living network alliance (DLNA) home networked device interoperability
guidelines –
Part 1: Architecture and protocols**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE

XH

CONTENTS

FOREWORD.....	7
0 Introduction	9
0.1 General.....	9
0.2 Purpose	9
0.3 Audience	10
0.4 Organization.....	10
1 Scope.....	12
2 Normative references	12
3 Terms, definitions and acronyms	13
3.1 Terms and definitions	13
3.2 Acronyms	29
4 DLNA home network architecture	30
4.1 General.....	30
4.2 Networking and connectivity	30
4.2.1 General	30
4.2.2 Network quality of service.....	31
4.3 Device discovery and control.....	31
4.4 Media management.....	31
4.5 Media formats	32
4.6 Media transport	32
5 DLNA device model.....	32
5.1 Overview	32
5.2 Device model elements	33
5.3 Device functions.....	34
5.4 Device categories.....	35
5.5 Device classes and roles.....	35
5.6 Device capabilities and roles	36
5.7 System usages.....	36
5.7.1 Overview	36
5.7.2 2-box pull system usage	37
5.7.3 2-box push system usage	38
5.7.4 3-box system usage.....	39
5.7.5 2-box printing system usage	40
5.7.6 3-box printing system usage	41
5.7.7 Download system usage	42
5.7.8 Upload system usage	43
5.8 Home infrastructure device (HID) system usage	44
5.8.1 General	44
5.8.2 Bridging HND and MHD network connectivity	44
5.8.3 Bridging HND and MHD media formats	45
5.9 Interoperability guidelines usage	46
6 Guideline terminology and conventions	48
6.1 Guideline compliance classifiers.....	48
6.2 Standard or specification usage classifiers	49
6.3 Guideline font usage conventions	49

6.4	Guideline syntax notation conventions	49
6.5	Guideline normative and informative text conventions	50
6.6	DLNA XML namespaces and schemas	50
7	Guideline requirements	50
7.1	General	50
7.2	Networking and connectivity	52
7.2.1	General	52
7.2.2	Normative definitions of NC-PS modes	53
7.2.3	Networking and connectivity – General capability requirements	53
7.2.4	Networking and connectivity – QoS requirements	62
7.2.5	Networking and connectivity – Device requirements	66
7.3	Device discovery and control	86
7.3.1	General	86
7.3.2	DDC UPnP device architecture	86
7.3.3	DDC UPnP Auto IP support	88
7.3.4	DDC UPnP SSDP default port	88
7.3.5	DDC UPnP discovery robustness	89
7.3.6	DDC UPnP HTTP support and general rules	92
7.3.7	DDC UPnP HTTP/1.0 rules	95
7.3.8	DDC UPnP HTTP/1.1 transaction rules	96
7.3.9	DDC UPnP HTTP persistent connections	97
7.3.10	DDC UPnP device responsiveness	98
7.3.11	DDC UPnP device description rules	99
7.3.12	DDC UPnP embedded device support	102
7.3.13	DDC UPnP service description rules	103
7.3.14	DDC UPnP XML namespace	104
7.3.15	DDC UPnP action argument encoding	104
7.3.16	DDC UPnP SOAP packet size	105
7.3.17	DDC UPnP error codes	105
7.3.18	DDC UPnP GENA packet size	106
7.3.19	DDC UPnP subscription handling	106
7.3.20	DDC UPnP UUID format	107
7.3.21	DDC UPnP UUID generation	107
7.3.22	DDC UPnP event subscription renewals	107
7.3.23	DDC UPnP event notification handling	108
7.3.24	DDC UPnP unknown header/tag/field robustness rule	108
7.3.25	DDC URI rules	109
7.3.26	DDC UPnP device description usage	111
7.3.27	DDC UPnP UDN usage	112
7.3.28	DDC UPnP multi-homing rules	113
7.3.29	DDC UPnP device icons	114
7.3.30	DDC UPnP UTF-8 support	115
7.3.31	DDC UPnP XML comments	115
7.3.32	DDC UPnP boolean types	115
7.3.33	DDC CP versioning	116
7.3.34	DDC absolute and relative URI requests	117
7.3.35	DDC maximum HTTP header size	117
7.3.36	DDC device capabilities	117
7.3.37	DDC DLNAQOS support	118

7.4	Media management	118
7.4.1	AV media management.....	118
7.4.2	Image printing media management	250
7.5	Media transport	255
7.5.1	General	255
7.5.2	Uniform client data availability model.....	258
7.5.3	Media operations	260
7.5.4	Media transport protocols	261
7.5.5	HTTP transport.....	274
7.5.6	RTP media transport.....	340
7.5.7	RTP media transport – Adaptation of media format profiles	363
7.5.8	RTP media transport – RTSP for control of RTP streams.....	386
7.6	Content transformation device virtualization	421
7.6.1	Theory of operations.....	421
7.6.2	Virtual device implementation	423
7.6.3	Virtual device, device discovery and control (DDC).....	424
7.6.4	Virtual device media management (MM).....	427
7.6.5	Virtual device media formats (MF)	438
7.6.6	Virtual device media transport (MT).....	439
7.7	Media interoperability unit (MIU).....	439
7.7.1	General	439
7.7.2	Media interoperability unit media management guidelines	440
Annex A (informative)	Network infrastructure device (NID) recommendations	443
Annex B (informative)	Tuner representation.....	449
Annex C (informative)	UPnP devices with multiple network interfaces.....	453
Annex D (informative)	Printer support	458
Annex E (informative)	Example applications of the uniform client data availability model (UCDAM).....	464
Annex F (informative)	Auto-IP developer guidance.....	469
Annex G (informative)	Mobile network connectivity and power-saving operation principles.....	475
Annex H (informative)	RTP protocol stack and SDP/RTSP/RTCP parameters	479
Bibliography.....		481
Figure 1 – DLNA functional components		30
Figure 2 – DLNA device model terms hierarchy.....		34
Figure 3 – 2-box pull system usage interaction model		38
Figure 4 – 2-box push system usage interaction model		39
Figure 5 – 3-box system usage interaction model		40
Figure 6 – 2-box printing system usage interaction model		41
Figure 7 – 3-box printing system usage interaction model.....		42
Figure 8 – Download system usage interaction model.....		43
Figure 9 – Upload system usage interaction model		43
Figure 10 – 2-box pull system usage interaction model between device categories.....		44

Figure 11 – M-NCF bridging the network connectivity gap between MHD and HND device categories.....	45
Figure 12 – Media interoperability between device categories.....	46
Figure 13 – Guideline layout and definitions	51
Figure 14 – Visual map of possible values for the attribute tables	52
Figure 15 – DLNA QoS visual organization	62
Figure 16 – UPnP discovery robustness.....	91
Figure 17 – Example of MM DLNA PlayContainer URI	192
Figure 18 – UCDAM summary.....	259
Figure 19 – Example of a valid and invalid pipelined POST transaction.....	339
Figure 20 – Calculated line	353
Figure 21 – Wall clock time sample accuracy distribution.....	354
Figure 22 – Packet with wall clock time sample header extension.....	356
Figure 23 – Example of packet with another header extension following wall clock time sample.....	356
Figure 24 – BFR packet format	363
Figure 25 – Content transformation with a virtual media server.....	422
Figure 26 – Content transformation with a virtual media renderer.....	423
Figure C.1 – UPnP device representation	453
Figure C.2 – UPnP device on multiple networks.....	454
Figure C.3 – Representation at the CDS level.....	455
Figure C.4 – Content URLs over multiple networks	456
Figure D.1 – Photo layout options	461
Figure D.2 – DMPr architecture components.....	461
Figure E.1 – Abstract representation of a stream	464
Figure E.2 – Stored content stream.....	465
Figure E.3 – Stream with no random access support.....	465
Figure E.4 – Stream with random access support	465
Figure E.5 – Live stream with growing buffer and no random access	466
Figure E.6 – Live stream with growing buffer and random access	466
Figure E.7 – Live stream with sliding buffer and random access support.....	466
Figure E.8 – Time-delayed live stream with sliding buffer and random access support.....	466
Figure F.1 – IP mixed network (Auto-IP and DHCP).....	469
Figure F.2 – Communication in mixed IP network.....	471
Figure F.3 – New routes in address transition flow.....	474
Figure G.1 – Illustration of the abstraction introduced by the NC-PS modes.....	477
Figure G.2 – NC-PS mode transition diagram	478
Figure H.1 – Overview of the protocol stack for RTP transport.....	479
Figure H.2 – SDP and RTSP parameters	479
Figure H.3 – RTCP parameters.....	480
Table 1 – Interoperable components	10
Table 2 – DLNA device classes in the HND device category	46
Table 3 – DLNA device capabilities.....	47

Table 4 – DLNA device classes in the MHD device category.....	47
Table 5 – DLNA device classes in the HID device category.....	48
Table 6 – DLNA namespace values	50
Table 7 – Normative definitions of network connectivity power saving (NC-PS) modes	53
Table 8 – Normative priorities for DLNA traffic types.....	63
Table 9 – BT-802.11 DLNAQOS access category mapping	84
Table 10 – IEEE 802.1D user priority values.....	85
Table 11 – Colour depth of device icons	114
Table 12 – CDS and UPnP maximum byte length.....	123
Table 13 – Namespace prefixes.....	129
Table 14 – Recommended metadata properties	129
Table 15 – CDS: Search minimum support of operators	175
Table 16 – UPnP: Class for searching all CDS objects.....	176
Table 17 – Capability ID syntax	185
Table 18 – Capability IDs for AnyContainer support.....	219
Table 19 – Required media class UPnP values	224
Table 20 – Required UPnP createClass elements.....	229
Table 21 – UPnP Printer dlna:X_DLNACAP element.....	252
Table 22 – DLNA media transfer modes.....	256
Table 23 – Permitted combinations of DLNAQOS, UP and transfer mode per media class.....	257
Table 24 – DLNA streaming media operation definitions	260
Table 25 – MT media class transfer modes.....	262
Table 26 – TTP prohibited operations references.....	323
Table A.1 – NID functions.....	443
Table A.2 – WMM access category mapping.....	446
Table A.3 – WMM Access and IEEE 802.1D priority.....	447
Table D.1 – DMPr printer verses PC attached printer.....	459
Table D.2 – Printing controller (+PR1+, +PR2+) UI components	460
Table D.3 – Printer status – Response.....	460
Table D.4 – UPnP PrintEnhanced:1 actions summary	462
Table D.5 – Evented variables	463
Table F.1 – Auto-IP Route	470
Table F.2 – DHCP Route	471
Table F.3 – Windows routing table example for device w/DHCP address	472
Table F.4 – Windows routing table example for device w/Auto-IP address	473
Table F.5 – Linux routing table example for device w/DHCP address.....	473
Table F.6 – Linux routing table example for device w/Auto-IP address.....	474
Table G.1 – Dynamic behaviour of the M-NCF depending on the current NC-PS mode	478

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**DIGITAL LIVING NETWORK ALLIANCE (DLNA) HOME NETWORKED
DEVICE INTEROPERABILITY GUIDELINES –****Part 1: Architecture and protocols****FOREWORD**

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC 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 National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication must be clearly indicated in the latter.
- 5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability must attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees 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, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) 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.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC must not be held responsible for identifying any or all such patent rights.

International Standard IEC 62481-1 has been prepared by IEC technical committee 100: Audio, video and multimedia systems and equipment.

The text of this standard is based on the following documents:

CDV	Report on voting
100/1127/CDV	100/1213/RVC

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts of the IEC 62481 series, published under the general title *Digital living network alliance (DLNA) home networked device interoperability guidelines*, can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the maintenance result date indicated on the IEC web site under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

Withdrawn
IECNORM.COM : Click to view the full PDF of IEC 62481-1:2007

0 Introduction

0.1 General

Consumers are acquiring, viewing, and managing an increasing amount of digital media (photos, music, and video) on devices in the consumer electronics (CE), mobile device, and personal computer (PC) domains. Consumers want to conveniently enjoy that content – regardless of the source – across different devices and locations in their homes. The digital home vision integrates the Internet, mobile, and broadcast networks through a seamless, interoperable network, which will provide a unique opportunity for manufacturers and consumers alike. In order to deliver on this vision, it was recognized that a common set of industry design guidelines would be required to allow companies to participate in a growing marketplace, leading to more innovation, simplicity, and value for consumers.

The digital living network alliance (DLNA) answered this challenge by taking the initiative to develop a workable framework for interoperable product design. The DLNA home networked device interoperability guidelines (hereinafter referred to as the interoperability guidelines) has been created in a unique cross-industry effort that combined the efforts of over 100 CE, PC-industry and mobile device companies from around the world who worked together with the aim of achieving the world's first substantial platform for true interoperability between personal computer and consumer electronic devices. The interoperability guidelines provide product developers with a long-term architectural view, plus specific guidance for IP-networked platforms, devices and applications in the home. The interoperability guidelines will be introduced in phases over several years to accompany the market adoption of usages and the availability of needed technology and standards.

The interoperability guidelines of this part of IEC 62481 include several informative annexes: Annex A, Annex B, Annex C, Annex D, Annex E, Annex F and Annex G.

The media formats contain

- a) ASF recommended procedures (informative);
- b) IFO file format (normative).

0.2 Purpose

The interoperability guidelines consists of two volumes covering architecture and protocols and media formats. It provides vendors with the information needed to build interoperable networked platforms and devices for the digital home. The necessary standards and technologies are available now to enable products to be built for networked entertainment centric usages. However, standards and technologies need to be clarified and options limited to ensure interoperability. The interoperability guidelines fulfil that role.

The interoperability guidelines are based on an architecture (see Clause 4) that defines interoperable components for devices and software infrastructure. It covers physical media, network transports, device discovery and control, media management and control, media formats, and media transport protocols.

Table 1 – Interoperable components

Key technology ingredients	
Functional components	Technology ingredients
Connectivity	Ethernet, IEEE 802.11 and Bluetooth
Networking	IPv4 suite
Device discovery and control	UPnP* Device Architecture v1.0
Media management and control	UPnP AV v1 and UPnP Printer:1
Media formats	Required and optional format profiles
Media transport	HTTP (mandatory) and RTP (optional)

0.3 Audience

The interoperability guidelines are intended for the following audiences:

- marketing professionals who specify requirements for home networked media products;
- developers who design and build home networked media products;
- quality assurance personnel who test and validate home networked media products.

0.4 Organization

This part of the interoperability guidelines is organized as follows. All annexes are informative.

Clause 2 Normative references: Information on ISO, IEC and recognized normative references contained in this standard. Other informative documents are listed in the bibliography.

Clause 3 Terms and acronyms: Definitions of terms and acronyms used in this standard.

Clause 4 DLNA home network architecture: An overview of the DLNA home networking architecture.

Clause 5 DLNA device model: An overview of the major device categories used to group guideline requirements.

Clause 6 Guideline terminology and conventions: Definitions for the compliance and usage classifications used for guideline requirements.

Clause 7 Guideline requirements: Covers guideline requirements for DLNA devices excluding media formats which are covered in another part of IEC 62481.

Annex A (informative) Network infrastructure device (NID) recommendations: Covers a set of recommendations for home network infrastructure devices such as gateways, routers, and hubs to ensure they work well with DLNA devices.

Annex B (informative) Tuner representation: Describes the way DLNA devices should represent tuner-based content.

Annex C (informative) UPnP devices with multiple network interfaces: Describes how a DLNA device can represent itself on multiple network interfaces. The annex also discusses how a content source should expose content URI values for different network interfaces.

Annex D (informative) Printer support: Introduces developers to the technical considerations required to support printers and also discusses some of the usability aspects of printing that are important for a good user experience.

Annex E (informative) Example applications of the uniform client data availability model: Clarifies the general applicability of the uniform client data availability model (UCDAM). It describes the data accessibility assumptions for both content sources and content receivers. The UCDAM model strives for completeness by using examples derived from stored, converted, and live content streams. The model also accounts for caching of data by content receivers.

Annex F (informative) Auto-IP developer guidance: Provides guidance for developers on extending Auto-IP support for IP stacks that have problems with full conformance to Auto-IP.

Annex G (informative) Mobile network connectivity and power saving operation principles: Provides guidance on network connectivity for mobile devices, including Bluetooth security and NC power-saving modes.

Annex H (informative) RTP protocol stack and SDP/RTSP/RTCP parameters: Provides graphic layout of the protocol stack for the RTP transport and SDP/RTSP/RTCP parameters.

DIGITAL LIVING NETWORK ALLIANCE (DLNA) HOME NETWORKED DEVICE INTEROPERABILITY GUIDELINES –

Part 1: Architecture and protocols

1 Scope

This part of IEC 62481 specifies the information needed to build interoperable networked platforms and devices for the digital home on audio, video and multimedia systems, including interoperable components for devices and software infrastructure, physical media, network transports, device discovery and control, media management and control, media formats, and media transport protocols. This standard also provides product developers with a long-term architectural view, plus specific guidance for IP-networked platforms, devices and applications in the home.

2 Normative references

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

IEC 62481-2, *Digital living network alliance (DLNA) home networked device interoperability guidelines – Part 2: Media formats*

ISO/IEC 13818-1:2000, *Information technology – Generic coding of moving pictures and associated audio information: Systems*

ISO/IEC 13818-9:1996, *Information technology – Generic coding of moving pictures and associated audio information – Part 9: Extension for real time interface for systems decoders*

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

ISO 10646:2003, *Information technology – Universal Multiple-Octet Coded Character Set (UCS)*

IEEE 802.1D:2004, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – IEEE standard for local and metropolitan area networks – Common specifications – Media access control (MAC) Bridges*

IEEE 802.1Q:2003, *IEEE Standard for Information Technology – Telecommunications and information exchange between systems – IEEE standard for local and metropolitan area networks – Common specifications – Virtual Bridged Local Area Networks*

IEEE 802.3:2002, *IEEE Standard for information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specification*

IEEE 802, *IEEE Standard for information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*

3 Terms, definitions and acronyms

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

3.1 Terms and definitions

3.1.1

AC 3

audio format standard for delivering up to 5.1 audio channels developed by Dolby Laboratories popularly known as Dolby Digital

3.1.2

access point

AP

device specially configured as a network infrastructure device on a wireless local area network (WLAN)

NOTE Access points act as a central transmitter and receiver of WLAN radio signals. APs used in home networks are generally small, dedicated hardware devices featuring a built-in network adapter, antenna, and radio transmitter. These APs support Wi-Fi wireless communication standards.

3.1.3

acknowledge

ACK

term typically used to describe an action following a network packet being successfully received

3.1.4

address resolution protocol

ARP

protocol in the TCP/IP family that resolves an IP address to a hardware address, such as an Ethernet address

3.1.5

advanced television systems committee

ATSC

one of the standard bodies for digital television broadcasting

3.1.6

application data unit

ADU

unit of data used for the RTP media transport

NOTE The definition of an ADU is different for each media stream. For audio media streams, an ADU is typically an audio frame. For video media streams, an ADU is typically a "slice" (for example, an NAL unit) or, in some cases, a complete video picture. Also, as a special case when MPEG-2 TS encapsulation is used, each TS packet is an ADU.

3.1.7

audio/visual profile

AVP

profile used for the RTP media transport

3.1.8

authentication

process by which an entity verifies that another entity is who or what it claims to be (examples of entities are: device, person, process.)

NOTE Authentication for Bluetooth is defined in Specification of the Bluetooth System (Volume 1 – Core, Volume 2 – Profiles, Addendum, Errata), Specification of the Bluetooth System, version 1.1, Bluetooth SIG, February 22, 2001, as the process of verifying which device is at the other end of the link. Authentication is

performed for devices based on their Bluetooth Address (BD_ADDR). In Bluetooth this is achieved by the authentication procedure based on the mutually stored link key or by pairing.

3.1.9

authorization

process of access check to determine whether the authenticated entity has access to the resource

NOTE Authorization for Bluetooth is defined in Specification of the Bluetooth System (Volume 1 – Core, Volume 2 – Profiles, Addendum, Errata), Specification of the Bluetooth System, version 1.1, Bluetooth SIG, February 22, 2001, as the process of deciding if a device is allowed to have access to a particular service. This is where the concept of “trusted” exists. Trusted devices (authenticated and indicated as “trusted”) are allowed access to services. Untrusted or unknown devices may require authorization based on user interaction before access to services is granted. This does not exclude the case where the authorization might be given by an application automatically. Authorization for Bluetooth always includes authentication.

3.1.10

extended audio/visual profile for RTCP-based feedback

AVPF

profile used for the RTP media transport

3.1.11

AVTransport:1 Service

AVT

UPnP service that provides network-based control for common transport operations such as play, stop, pause, next, previous, and seek, a standard UPnP DCP

3.1.12

background user priority

BK

priority-based QoS level for background user priority

3.1.13

bearer

physical and link-level network transport, such as Bluetooth or IEEE 802.11

3.1.14

best-effort user priority

BE

priority-based QoS level for best-effort user priority

3.1.15

bluetooth device address

BD_ADDR

unique 48-bit bluetooth hardware address

3.1.16

Bluetooth network encapsulation protocol

BNEP

bluetooth PAN profile concept

3.1.17

cacheable content

content binaries whose binary representations remain static over time

NOTE By default, HTTP allows intermediate HTTP caches to store such items and respond to similar HTTP requests as a means to accelerate the interaction with the user. In the interoperability guidelines cacheable content includes (but it is not limited to) : Image, audio, AV content that exists as stored files. Content resulting from transcoding or encoding operations when the output binaries can be preserved over time.

3.1.18**canonical name****CNAME**

canonical endpoint identifier used to associate streams and differentiate sources in RTP media transport

3.1.19**channel**

channel refers to one or more media streams that together constitute a unique entity for the purpose of announcement, selection, and rendering

NOTE For example, for digital television sources, a channel is equivalent to an ATSC "virtual channel", a DVB "service", or an MPEG-2 "program". For digital radio sources, a channel is equivalent to a single "station".

3.1.20**coded data buffer****CDB**

data buffer used for the RTP media transport, buffer space that is used to store compressed data before decoding

3.1.21**ConnectionManager:1 Service****CMS**

UPnP service that provides information about the supported transport protocols and media formats of a UPnP device, a standard UPnP DCP

3.1.22**consumer electronics****CE**

class of devices used in the home, such as DVD, DVR, PVR, PDA, TV, set top box, cellular phones

3.1.23**content**

aggregation of one or more works

3.1.24**content binary**

binary representation of content for the purpose of storage or transfer over communication links

3.1.25**ContentDirectory:1 Service****CDS**

UPnP service that provides network-based discovery of content, a standard UPnP DCP

3.1.26**content receiver**

endpoint that consumes content received via a network transfer from another endpoint

3.1.27**content source**

endpoint that places content onto the network for transfer to another endpoint

3.1.28**content transformation**

transcoding, transrating, or scaling of a content binary

3.1.29

contributing source

CSRC

source used for the RTP media transport

3.1.30

Data – High Rates 1 through 5

DH1-5

bluetooth specific physical layer packet types

3.1.31

Data – Medium Rates 1 through 5

DM1-5

bluetooth specific physical layer packet types

3.1.32

decoder friendly point

point in the stream that the decoder can begin to process data without any other internal state information about the stream so that the decoder can begin processing at that point and create a valid output rendering

3.1.33

device architecture 1.0

DA

UPnP device architecture version 1.0 document

3.1.34

device capability

set of device functions (at least 1) aggregated to support a system usage

NOTE A device capability cannot stand alone and should be deployed in conjunction with an implementation of a valid DLNA device class. Since a device capability does not stand alone, it is not required to have components in all layers of the DLNA architecture. A device capability may have a one-to-one correspondence to a device function. A device capability is a certifiable entity only when it is implemented as an addition to at least one device class.

3.1.35

device category

group of device classes with the same environmental characteristics and sharing common system usages that are enabling home networking use case scenarios

NOTE Examples used within this standard are home network device (HND), mobile handheld device (MHD), and home infrastructure device (HID). While device classes are grouped within a device category, a single physical device may support device classes that fall into multiple device categories.

3.1.36

device class

class defined by a set of device functions. It specifies the features supported on a device regardless of its physical attributes

NOTE Examples used within this document are digital media server (DMS) and digital media player (DMP). A single device may support multiple device classes. A DLNA device must support at least one device class and may support one or more device capabilities. A device class is the certifiable entity in DLNA.

3.1.37

device control protocol

DCP

specification that is standardized by the UPnP forum

NOTE Related specifications produced by a UPnP working committee are often identified by the working committee name. For example, UPnP AV 1 DCP.

3.1.38**device discovery and control****DDC**

section heading in the interoperability guidelines that defines the underlying interoperability architecture for the discovery and control devices

3.1.39**device function**

non-decomposable operational property (for example, IP connectivity) which should be supported by existing standards

3.1.40**device option**

function which provides optional extensibility to existing device function requirements or a new optional device function to the DLNA architecture

3.1.41**device type**

specific device defined by its realization and its usage models, DVD players, TVs, DVRs, mobile phones, printers, cameras, picture frames, etc., which are used primarily for marketing descriptions, and should not be used in guideline definitions

3.1.42**differentiated services (DiffServ) code point****DSCP**

QoS field, defined by the DiffServ discipline, IETF RFC 2326, found in the layer 3 header of IP packets

3.1.43**digital item declaration language****DIDL**

XML schema for representing the metadata of digital content

3.1.44**Digital item declaration language – Lite****DIDL-Lite**

XML schema used by the UPnP Forum for representing the metadata of digital content using a subset of the DIDL schema with additional metadata properties defined by the UPnP Forum

3.1.45**digital media controller****DMC**

DLNA device class having home network environmental characteristics, with the role of finding content exposed by a DMS and matching it to the rendering capacities of a DMR and setting up the connections between the DMS and the DMR

3.1.46**digital media player****DMP**

DLNA device class having home network environmental characteristics, with the role of finding content exposed by a DMS and rendering the content locally

3.1.47**digital media printer****DMPr**

DLNA device class having home network environmental characteristics, with the role of providing document and image printing services to other DLNA devices

3.1.48

digital media renderer

DMR

DLNA device class having home network environmental characteristics, with the role of rendering content it receives after being setup by another network entity

3.1.49

digital media server

DMS

DLNA device class having home network environmental characteristics, with the role of exposing and distributing content throughout the home

3.1.50

digital video broadcasting

DVB

one of the standard bodies for digital television broadcasting

3.1.51

DLNA QoS user priority

DLNAQOS_UP

DLNA-defined QoS label used to correlate an underlying IEEE 802.1Q user priority and WMM access category to a DLNA traffic type(s)

3.1.52

domain name system

DNS

protocol that enables hierarchical names for Internet domains and addresses which includes the means to translate between numerical IP addresses and text host names

3.1.53

dynamic host configuration protocol

DHCP

protocol to automatically provide IP addresses and other network configuration information to network nodes

3.1.54

elementary stream

ES

single stream of several MPEG-2 carried components

3.1.55

exposed

content that is listed by a UPnP AV ContentDirectory service (CDS) and may not necessarily exist at the time that it is exposed (i.e. will require transcoding or conversion)

3.1.56

extensible markup language

XML

text-based declarative language used to describe structured data for information exchange

3.1.57

full TS

piece-wise constant rate MPEG-2 transport stream that is fully compliant with 2.4.2.2 of ISO/IEC 13818-1 which is characterized by a consistent temporal relationship (or "even spacing") between any two adjacent TS packets

3.1.58**high definition****HD**

picture quality at an HDTV level

3.1.59**high definition television****HDTV**

television system which provides a higher quality display, with a vertical resolution display from 720p to 1080i and higher and an aspect ratio (the width to height ratio of the screen) of 16:9, for a viewing experience similar to watching a movie

3.1.60**home infrastructure device****HID**

device category that groups together all the applicable DLNA device classes, which provide support for interoperability between device classes of different device categories, where device classes in this device category are: MNCf and MIU

3.1.61**home network device****HND**

device category that groups together all the applicable DLNA device classes with home network environmental characteristics (requirements), where device classes in this device category are: DMS, DMP, DMR, DMC, and DMPr

3.1.62**hyper text transfer protocol****HTTP**

protocol for transferring files across the Internet which requires an HTTP client program on one end, and an HTTP server program on the other end

3.1.63**ideal network conditions**

conditions of a network state used only for testing and validation of guideline compliance

NOTE The effective network capacity should be substantially greater than the aggregate bandwidth of the content under test, and there are no additional devices competing for available network resources at the time of test.

3.1.64**instance state variable**

new type of state variable defined by the AVTransport and RenderingControl services which are state variables associated with a virtual instance of a service

NOTE Both services use an evented state variable by the name of LastChange, to report a value change of an instance state variable.

3.1.65**interactive transfer**

media transfer mode where content that does not contain internal timing information is being transferred for the purpose of immediate user interaction with the content, for example, sending images that are to be displayed immediately to a user

3.1.66**Internet control message protocol****ICMP**

protocol in the TCP/IP family that is used for out-of-band messages related to network operation

3.1.67

Internet gateway device

IGD

multifunction network infrastructure device that routes and/or bridges global internet with the local area network

3.1.68

Internet protocol version 4

IPv4

OSI network layer 3 protocol

3.1.69

joint photographic experts group

JPEG

coding standard for compression of still images (pictures)

3.1.70

linear pulse code modulation

LPCM

uncompressed audio encoding

3.1.71

link-level bearer

bearer restricting the concept to include only the physical layer and link-layers as described for the ISO's OSI model

3.1.72

link management protocol

LMP

Bluetooth specific protocol

3.1.73

local area network

LAN

closely administered network segment(s) such as within the home or office

3.1.74

logical link control and adaptation protocol

L2CAP

bluetooth link layer protocol

3.1.75

low frequency enhancement

LFE

DVB-specified way to transmit additional sound information

3.1.76

media class

type of media a device type or device class supports

NOTE The media classes used in this document are image, audio only, and video with audio (AV).

3.1.77

media collection file

content binary whose purpose is to reference other content binaries usually for sequenced playback, often used for audio or AV playlists or image slideshows

3.1.78**media format**

format type for content of a media class that is exposed by a UPnP AV MediaServer contained in a device that acts as a DMS

NOTE Examples for the media classes are: Image – JPEG; Audio – LPCM; AV – MPEG-2.

3.1.79**media formats****MF**

section heading in the interoperability guidelines

3.1.80**media interoperability unit****MIU**

DLNA device class that provides media format interoperability between devices in the HND and MHD device categories

3.1.81**media management****MM**

section heading in the interoperability guidelines

3.1.82**MediaRenderer:1 Control Point****MRCP**

UPnP control point that issues actions to an MRD

3.1.83**MediaRenderer:1 Device****MRD**

UPnP device that provides network-based control for the rendering of content, where minimally a MediaRenderer must have a RenderingControl Service and a ConnectionManager service

NOTE The MediaRenderer specification is a standard UPnP DCP.

3.1.84**MediaServer:1 Control Point****MSCP**

UPnP AV control point that issues actions to an MSD

3.1.85**MediaServer:1 Device****MSD**

UPnP device that provides network-based discovery of content where, minimally, a MediaServer must have a ConnectionManager Service and a ContentDirectory Service.

NOTE The MediaServer specification is a standard UPnP DCP.

3.1.86**media stream**

single elementary stream or MPEG-2 TS or MPEG-2 PS multiplex as used for the RTP media transport

3.1.87**media transfer mode**

type of transfer used to deliver content from a content source to a content receiver

NOTE There are three types of DLNA media transfer modes: streaming transfer, interactive transfer and background transfer.

3.1.88

media transport

MT

section heading in the interoperability guidelines

3.1.89

mobile digital media controller

M-DMC

DLNA device class having mobile handheld environmental characteristics, with the role of finding content exposed by an M-DMS and matching it to the rendering capacities of a DMR and setting up the connections between the M-DMS and the DMR

3.1.90

mobile digital media downloader

M-DMD

DLNA device class having mobile handheld environmental characteristics, with the role of downloading content from an M-DMS

3.1.91

mobile digital media player

M-DMP

DLNA device class having mobile handheld environmental characteristics, with the role of finding content exposed by an M-DMS and rendering the content locally

3.1.92

mobile digital media server

M-DMS

DLNA device class having mobile handheld environmental characteristics, with the role of exposing and distributing content throughout the home

3.1.93

mobile digital media uploader

M-DMU

DLNA device class having mobile handheld environmental characteristics, with the role of uploading content to an M-DMS

3.1.94

mobile handheld device

MHD

device category that groups together all the applicable DLNA device classes with mobile handheld environmental characteristics (requirements), where device classes in this device category are: M-DMS, M-DMP, M-DMD, M-DMU, and M-DMC

3.1.95

mobile network connectivity function

MNCF

DLNA device class that provides interoperability by bridging the network connectivity layer between devices in the HND and MHD device categories

3.1.96

multimedia home platform

MHP

optional application interface used together with MPEG-2 transmissions

3.1.97**multipurpose Internet mail extension****MIME**

standard system for identifying the type of data contained in a file which is an Internet protocol that allows sending binary files across the Internet as attachments to e-mail messages including graphics, photos, sound, video files, and formatted text documents

3.1.98**national television systems committee****NTSC**

standard for broadcast and reception of analogue television signals

3.1.99**native device**

device creating a virtual device and adding functionality to an existing device in the network

NOTE Device can be server or renderer. These native devices are in contrast to a virtual server or renderer.

3.1.100**network abstraction layer unit****NAL unit**

term specific to H.264 RTP payload format which is a 1-byte header and the payload byte string (as defined in IETF RFC 3984, RTP Payload Format for H.264 video)

3.1.101**network access point****NAP**

Bluetooth PAN profile concept

3.1.102**networking and connectivity****NC**

section heading in the interoperability guidelines

3.1.103**network de-jitter buffer**

buffer space that is used to store data for de-jittering and de-interleaving as used for the RTP media transport which includes RTP header and payload

3.1.104**network infrastructure device****NID**

device which provides supporting functionality in the home network such as access points, bridges, Internet gateways, routers, and switches, but which is not a member of the normative HID device category

NOTE These devices facilitate a good user experience with DLNA devices but are only covered at this time in this standard by informative recommendations in Annex A.

3.1.105**non-cacheable content**

content binaries whose binary representations are valid only for one particular transaction at a particular instant of time

NOTE Intermediate HTTP caches need directives to prevent the default caching of such content. In these interoperability guidelines non-cacheable content includes (but it is not limited to): live TV streams. Content resulting from transcoding or encoding operations when the output binaries have been optimized for a particular transaction (for example, encoding to match the channel conditions for a given transaction).

3.1.106

open systems interconnection

OSI

networking stack model (7 layers)

3.1.107

PAN user

PANU

bluetooth PAN profile concept

3.1.108

partial SPTS

partial MPEG-2 transport stream which is also a single program transport stream (SPTS)

3.1.109

personal area network

PAN

one of the Bluetooth profiles

3.1.110

personal computer

PC

general-purpose computer equipped with a microprocessor and designed to run commercial software (such as a word processor or World Wide Web browser) for an individual user

3.1.111

phase alternating line

PAL

standard for broadcast and reception of analogue television signals

3.1.112

portable network graphics

PNG

coding standard for compression of still images (pictures)

3.1.113

post-decoder buffer

buffer space used to store decompressed data before rendering as used for the RTP media transport

3.1.114

pre-decoder buffer

hypothetical reference decoder buffer that is used to contain a media (audio/video) stream after it has arrived from the network and before it is decoded into a renderable frame as used for the RTP media transport

3.1.115

printer

device defined in UPnP Printer:1 Annex A V1.0, Printer:1 Device Template Version 1.01 with Annex A – Optional Service Addition V1.0

NOTE Printers are required to provide a printer service called PrintEnhanced and can optionally provide other services (for example, a fax service). In the case of the DLNA DMPv device class, the required print service is PrintEnhanced:1.

3.1.116

printer control point

PrCP

UPnP control point that issues actions to a PrD

3.1.117**printer device****PrD**

UPnP device that provides network-based control for the printing, which minimally must have a PrintEnhanced service

NOTE The printer device specification is a standard UPnP DCP.

3.1.118**printing controller**

DLNA device capability that contains a UPnP control point that allows the user to select the content they want to print, select the printer they want to use and monitor the status on the job.

NOTE The printing controller will be the component that composes the XHTML-print document that is then passed to the printer. A printing controller can be the source of Image content (+PR1+). The device capability acronyms used in these guidelines are +PR1+ and +PR2+.

3.1.119**program clock reference****PCR**

term defined by ISO/IEC 13818-1

3.1.120**program stream****PS**

content binary stream usually in reference to an MPEG-2 AV stream format

3.1.121**quality of service****QoS**

quality of service to provide guarantees on the ability of a network to deliver predictable results

3.1.122**real time protocol****RTP**

media transport that provides end-to-end network transport functions for transmitting real-time data, such as AV and provides services such as payload type identification, sequence numbering, time-stamping, and delivery monitoring

3.1.123**receiver buffer**

total buffer space used to store data received from the server before the decoding as used for the RTP media transport

3.1.124**receiving endpoint**

endpoint defined specifically through 7.5.6.2.4, which requires an RTP client and an RTSP client

3.1.125**renderingControl:1 Service****RCS**

UPnP service that provides network-based control for the adjustment of rendering attributes such as volume, brightness, contrast, and mute, a standard UPnP DCP

3.1.126

rendering endpoint

term used to denote any device class that receives content for playback or streaming scenarios, for example, the DMP or M-DMD device classes

3.1.127

RTP media transport

transport mechanism for real-time media streams between DLNA device classes and capabilities

NOTE This transport mechanism uses RTP, RTCP, RTSP, SDP protocols, and RTP payload formats with their associated media profiles.

3.1.128

RTP session

one or more RTP streams that are transmitted to the same destination IP address and UDP port

NOTE Typically, there is a one-to-one mapping between RTP streams and RTP sessions, but it is possible for multiple RTP streams to use the same RTP session (port multiplexing). Note that associated RTCP traffic is also part of that RTP session although the packets are sent to the next higher UDP port number.

3.1.129

RTP stream

media stream that is encapsulated in RTP

NOTE All of the RTP packets have the same SSRC and are transmitted on the same RTP session.

3.1.130

RTSP session

session typically consisting of a client creating one or more RTP Sessions (SETUP), starting the stream with PLAY or RECORD, and closing the RTSP Session with TEARDOWN, complete RTSP "transaction", for example, the viewing of a movie

NOTE RTSP sessions are identified using the RTSP "session" header.

3.1.131

scaling

changing the visual extent of an image or video portion of an AV media stream

3.1.132

service control protocol description

SCPD

XML-encoded file describing a UPnP service known as a service description file

3.1.133

serving endpoint

endpoint defined specifically through 7.5.6.2.3, which includes the RTP server and RTSP server

3.1.134

simple object access protocol

SOAP

XML-based messaging protocol used to exchange service requests and responses over a network

3.1.135

simple service discovery protocol

SSDP

UPnP device discovery protocol

3.1.136**streaming transfer**

term indicating that the packets are transferred minimally at a rate sufficient for real-time rendering for audio and AV streams

NOTE This definition does not imply that the receiving endpoint will always render content exchanged using streaming transfer (for example, it may store the content).

3.1.137**system usage**

usage which describes a device interaction model between device classes and/or device capabilities derived when enabling home networking use case scenarios

3.1.138**test conditions**

test conditions when all of the following conditions are satisfied:

- only 1 content receiver and only 1 content source;
- only 1 active connection at any given time

NOTE Both devices (content receiver and content source) should be configured by vendors before testing begins to have sufficient resources to successfully complete the stated test requirements. Devices interact with each other under ideal network conditions.

3.1.139**tagged image file format****TIFF**

media format of still images (pictures)

3.1.140**transcoding**

changing the coding system used for a content binary

3.1.141**transmission control protocol****TCP**

protocol in the TCP/IP family used for the reliable exchange of data over a network

3.1.142**transport stream****TS**

content binary stream usually in reference to an MPEG-2 AV stream format

3.1.143**transrating**

changing the rate or compression parameters used within the coding system for a content binary

3.1.144**uniform resource identifier****URI**

W3C's codification of the name and address syntax of present and future objects on the Internet

NOTE In its most basic form, a URI consists of a scheme name (such as file, http, ftp, news, mailto, gopher) followed by a colon, followed by a path whose nature is determined by the scheme that precedes it. URI is the umbrella term for URNs, URLs, and all other uniform resource identifiers.

3.1.145

UPnP

architecture for pervasive peer-to-peer network connectivity of devices of all form factors

NOTE It is designed to bring easy-to-use, flexible, standards-based connectivity to *ad hoc* or unmanaged networks whether in the home, in a small business, public spaces, or attached to the Internet. It is a distributed, open networking architecture that leverages TCP/IP and Web technologies to enable seamless proximity networking in addition to control and data transfer among networked devices in the home, office, and public spaces.

3.1.146

UPnP control point

CP

generic reference to any UPnP control point

3.1.147

user datagram protocol

UDP

protocol in the TCP/IP family used for the unreliable exchange of data over a network

3.1.148

user priority

UP

3-bit field of the QoS control field in the WMM Specification, Wi-Fi WMM (wireless multimedia) specification and a tag control information field in the VLAN tag header of IEEE 802.1Q-2003, that defines the relative priority of a packet

3.1.149

video user priority

VI

priority-based QoS level for video user priority

3.1.150

virtual instance

mechanism by which a UPnP device can have multiple instances of the same UPnP service

NOTE Control points that interact with the AVTransport and RenderingControl services should interact with a virtual instance of the service. Each virtual instance is identified by a non-negative instanceID value.

3.1.151

virtual renderer

DMR that accepts content and sends it to another DMR within the network for rendering which accepts a wider range of content types, formats, rates, or sizes than the native device

3.1.152

virtual server

DMS or M-DMS which exposes content existing on another DMS or M-DMS, possibly containing additional media types through content transformation

3.1.153

VLAN tag

field on a layer-2 packet header defined by IEEE 802.1Q-2003

3.1.154

voice user priority

VO

priority-based QoS level for voice user priority

3.1.155**wide area network****WAN**

usually in reference to the network outside the home or office typically in reference to the entire global Internet

3.1.156**Wi-Fi multimedia****WMM**

name used to describe the QoS guidelines and associated test plans published by the Wi-Fi Alliance for IEEE 802.11 wireless networks

3.1.157 **ΔT_{PCR}**

difference in time, measured in 90 kHz clock units, between successive PCR values in the TS stream as used for the RTP media transport, represented mathematically as

$$\Delta T_{\text{PCR}} = \text{PCR}_{n+1} - \text{PCR}_n \quad (1)$$

NOTE This value is utilized in timestamp equations in the "Guidelines for encapsulation of MPEG-2 Transport streams".

3.2 Acronyms

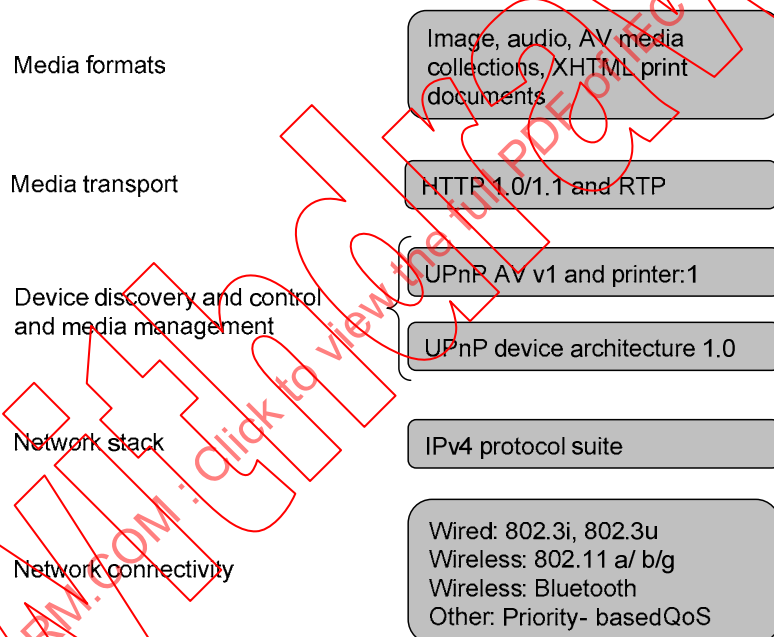
AV	Audio with Video
BT	Bluetooth
DLNAQOS	DLNA-defined QoS model
DVD	Digital Versatile Disc
DVR	Digital Video Recorder
IP	Internet Protocol
IPR	Intellectual Property Rights
MPEG-2	Moving Picture Experts Group phase 2
NC-PS	Network Connectivity Power Saving
PVR	Personal Video Recorder
RTCP	Real Time Control Protocol
RTSP	Real Time Streaming Protocol
SDES	Session Description Item
SDP	Session Description Protocol
SPTS	Single Program Transport Stream
SSRC	Synchronization Source
TTS	Timestamped Transport Stream

URL	Uniform Resource Locator
UTF	Unicode Transformation Format
VLAN	Virtual Local Area Network

4 DLNA home network architecture

4.1 General

To achieve interoperability between connected digital media devices in the home, a common set of building blocks based on existing standards is needed as a basis to develop the interoperability guidelines. Table 1 shows the specific functional components and technology ingredients that are covered in the interoperability guidelines. Figure 1 illustrates these functional components within the networking architecture of the interoperability guidelines. The interoperability guidelines define usage of these functional components to ensure interoperability among device classes defined in Clause 5. A brief overview of each functional component follows in the subsequent subclauses.



IEC 1569/07

Figure 1 – DLNA functional components

4.2 Networking and connectivity

4.2.1 General

The IPv4 protocol suite is the foundation for networking and connectivity for DLNA devices in the digital home. IP also provides the underlying network communications for applications on the Internet. Based on industry-standard specifications from the IETF, IP is implemented and supported in a wide range of devices. IP has the following advantages for use by DLNA devices.

- IP has demonstrated that it allows applications to run over different network topologies transparently.
- IP allows connecting every device in the home to the Internet.
- IP connectivity solutions are widely used and are cost-effective. The most common ones are Ethernet (802.3i and 802.3u) and wireless technologies (802.11a, 802.11b, and

802.11g) for devices in the home networking environment. In the mobile handheld environment, Bluetooth is the prevalent wireless technology in use.

Subclause 7.2 specifies the detailed guidelines to enable interoperability between DLNA devices in the digital home. In addition, the home environment requires supporting network infrastructure, such as access points, bridges, Internet gateways, routers, and switches. These non-normative devices are referred to in this standard as network infrastructure devices (NID). Annex A provides informative recommendations for network infrastructure devices to facilitate a good user experience and interoperability with DLNA devices.

4.2.2 Network quality of service

Multimedia applications on IP networks benefit from quality of service (QoS) functionality to optimize the way shared network resources are allocated among different applications. Without QoS, all applications running on different devices have an equal opportunity to transmit data frames. Multimedia applications such as video streaming and music streaming are sensitive to excessive latency variations and throughput reductions. With prioritized QoS, applications label (tag) packets to indicate the user priority (UP) that dictates how the packets are allowed to access network resources.

The DLNA QoS model is intended to allow DLNA applications that wish to take advantage of user priority to have common usage rules for tagging. Devices that do not wish to use QoS must be tolerant of tagging. The DLNA QoS model promotes fair and consistent usage of priorities and balanced performance across all DLNA traffic types, in addition to interoperability, thus enhancing the overall user experience.

4.3 Device discovery and control

Device discovery and control enables a device on the home network to discover the presence and capabilities of other devices on the network and collaborate with these devices in a uniform and consistent manner. The UPnP device architecture, version 1.0, addresses all of these needs and simplifies device networking in the home. For this reason, UPnP device architecture is the device discovery and control solution for DLNA devices. Subclause 7.3 specifies the detailed guidelines to enable interoperability between DLNA devices in the digital home.

4.4 Media management

Media management enables devices and applications to identify, manage, and distribute media content across the home network devices. UPnP audio/video (AV) and UPnP printer technology addresses all of these needs for the home network and is the media management solution for DLNA devices.

The UPnP AV architecture defines the interaction model between UPnP AV devices and associated control point applications. UPnP AV devices can instantiate themselves in a variety of form factors, including (but not limited to) TVs, VCRs, DVD players, set-top boxes, stereo systems, still-image cameras, portable media players, cell phones, and PCs. The UPnP AV architecture allows devices to support entertainment content in any format using any media transfer protocol. The UPnP AV specification defines two types of UPnP devices on the home network: UPnP AV MediaServers and UPnP AV MediaRenderers. The specifications also define four services hosted by UPnP AV MediaServers and UPnP AV MediaRenderers. The existence of UPnP control points that interact with UPnP AV devices and services is implied.

- a) Content directory service: Exposes the available content.
- b) Connection manager service: Determines how the content can be transferred from the UPnP AV MediaServer to the UPnP AV MediaRenderer devices.
- c) AV transport service: Controls the flow of the content.
- d) Rendering control service: Controls how the content is played.

The UPnP printer architecture defines the interaction model between UPnP printing devices and associated control point applications. Examples of UPnP printer devices are photo printers. The UPnP printer architecture defines a UPnP printer device which specifies the UPnP PrintEnhanced:1 service which in turn specifies XHTML-print, CSS print, and CSS print enhanced layout extension as the page description language.

See Clause 5 for further information on how UPnP technology components are mapped into DLNA device classes.

Subclause 7.4 specifies the detailed guidelines to enable interoperability between DLNA devices in the digital home.

4.5 Media formats

Media formats describe how content is encoded and formatted for transport and rendering on the home network. The DLNA media format model is intended to achieve a baseline for network interoperability while encouraging continued innovation in media codec technology. For each device category, the DLNA media format model defines a set of mandatory and optional media format profiles for each of the three classes of media: imaging, audio, and AV. A media format profile is a set of attributes, parameters, and system and compression level details sufficient to describe the media format of a content binary to enable interoperability between DLNA devices in each device category. In order to support interoperability between devices of different device categories, the media interoperability unit performs basic translation between the required media format profiles of different device categories. In addition, the DLNA media format model specifies rules about conversion between optional and mandatory formats to ensure that content can be enjoyed on all devices. IEC 62481-2 specifies the detailed guidelines to enable interoperability between DLNA devices in the digital home.

4.6 Media transport

Media transport defines how content travels across the home network. DLNA devices that source or receive media content across the home network must support HTTP as the baseline transport mechanism for the transfer of content. In addition, the RTP transport can optionally be used as a media transport, but the mandatory requirements for HTTP must always be supported. Subclause 7.5 specifies the detailed guidelines to enable interoperability between DLNA devices in the digital home.

5 DLNA device model

5.1 Overview

The interoperability guidelines address the requirements of devices with differing environmental characteristics, such as home network and mobile handheld devices. Home network devices (HNDs) and mobile handheld devices (MHDs) are device categories that have a differing set of requirements in media formats and network connectivity. This clause provides a device model with consistent terms and usages for these device categories. To support interoperability between home network devices and mobile handheld devices, it is possible for a home network device to meet all the requirements for the corresponding mobile handheld device. It is also possible for a mobile handheld device to meet all the requirements for the corresponding home network device. In these cases, such a device is a member of both the HND and a MHD device categories; but, in most cases, that may not be feasible, so another way to achieve interoperability is via a group of devices that will be able to provide bridging or content transformation services between these two device categories. These devices belong to a device category referred to as the home infrastructure device (HID). The following summarizes these devices.

- Bridging network connectivity between the MHD and the HND device categories (7.2).
- Media format interoperability services between the MHD and the HND device categories (7.7).

In summary, the key points about device categories are as follows.

- Each is uniquely optimized for the requirements of a particular environment.
- The device guidelines focus on interoperability of devices within a device category.
- There are guidelines for devices which facilitate interoperability between device categories.
- A device may choose to be a member of multiple device categories.

5.2 Device model elements

As described in Clause 4, devices adhering to the interoperability guidelines have six architectural layers. In summary, they are media formats for describing conformant content, media management for describing how content is found and controlled to achieve different system usages, device discovery and control for device control, media transport for the transfer of content, network stack for IPv4 protocol requirements, and network connectivity for supporting different network physical layers. The following are terms used within this clause and throughout the guidelines. Their interdependence is illustrated in Figure 2.

A device category is an aggregation of device classes with common environmental characteristics (for example, mobile devices) and sharing system usages that enable home networking use case scenarios. An example of a device category is the set of all device classes with system usages that solve requirements in media formats and network connectivity in a home network environment, such as an HND. Device classes are grouped within a device category, but a single physical device may fall into multiple device categories.

A system usage describes a device interaction model between device classes and/or device capabilities. System usages are derived when enabling home networking use case scenarios. An example is a rendering device with a dedicated user interface that is browsing, selecting, and playing content from a media server on the home network, such as the 2-box pull system usage.

A device class is a set of device functions (at least one) aggregated to be used in a system usage that enables home networking AV use case scenarios. A device class must provide support for all layers in the DLNA architecture except when defined within the home infrastructure device category which is providing interoperability between one or more layers in the DLNA architecture. A device class is a certifiable entity by DLNA and is derived from system usages. It specifies the capabilities supported by a device regardless of the device's physical attributes. An example of a device class is a device with the role of exposing and distributing content throughout the home such as a DMS. A single physical device may support multiple device classes.

A device capability is a set of device functions (at least one) aggregated to be used in a system usage that is enabling home networking AV use case scenarios. A device capability does not provide support for all layers in the DLNA architecture. It typically contains device functions at the device discovery and control, media management, and media transport layers only. A device capability is not a device class and cannot stand alone. It must always be deployed in conjunction with an implementation of a valid device class. A device class may already contain some of the device functions required to provide a device capability. An example of a device capability is any DLNA device that incorporates the additional feature (capability) of pushing content to a rendering device, such as a push controller.

A device function is a non-decomposable operational property. Device functions should be supported by existing standards or specifications. A device function usually applies to a single layer within the DLNA architecture. An example of a device function is an operational

component at the device discovery and control layer of the DLNA architecture such as a UPnP device. Device classes and device capabilities are composed of a set of device functions. Device functions are the building blocks of DLNA devices.

A device option provides optional extensibility to an existing device class definition, such as upload functionality added to a MediaServer device (MSD), or it provides a new optional device function to the DLNA architecture, such as RTP. A device option differs from a device class or device capability in that it normally enables a device class or device capability to perform an existing system usage in a different way (for example, RTP). In the case where a device option achieves a new system usage, it adds functionality to an existing device class or device capability to support a new interaction such as the upload system usage.

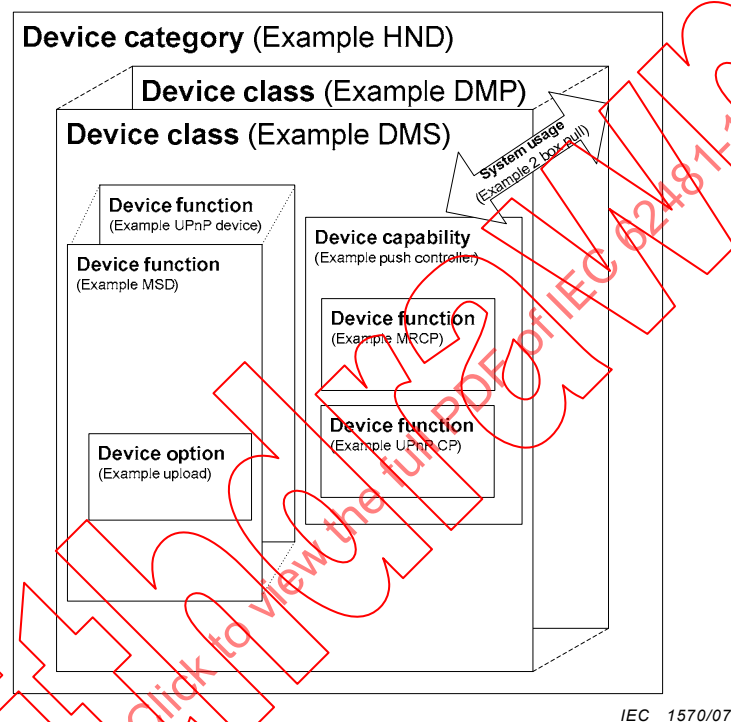


Figure 2 – DLNA device model terms hierarchy

5.3 Device functions

For the interoperability guidelines and system usages, the device functions below are defined for the DLNA architecture.

- **IP connectivity** – Network connectivity and network stack (see 7.2). This incorporates Ethernet (802.3), 802.11, and/or Bluetooth connectivity and IP networking using the IPv4 protocol suite.
- **UPnP device** and **UPnP control point (UPnP CP)** – Device discovery and control based upon the UPnP Device Architecture (see 7.3). This incorporates the baseline device architecture used by all device classes and device capabilities.
- **UPnP AV MediaServers (MSD), UPnP AV MediaServer control point (MSCP), UPnP AV MediaRenderer (MRD), UPnP AV MediaRenderer control point (MRCP), UPnP printer device (PrD), and UPnP printer control point (PrCP)** – Media management (see 7.4). This incorporates the control functionality that is layered on top a UPnP Device or a UPnP control point to fulfil a role for a device class or a device capability in a system usage. An MSD provides methods to access to content. An MSCP is a controller that can browse and select content provided by an MSD. An MRD provide methods to render content. An MRCP is a controller that selects the content to be rendered by an MRD. A PrD provides the ability to print image content. A PrCP is a controller that creates print jobs for selected content to be printed by a PrD.

- **Media transport server and media transport client** – Media transport (see 7.5). These are the device functions for the transport of content. The mandatory transport for content is HTTP which has the components of an HTTP server and an HTTP client. An optional transport for content is RTP which has the components of an RTP serving endpoint and an RTP receiving endpoint. RTP is an example of a device option which provides optional extensibility to system usages utilizing a media transport.
- **Content** – IEC 62481-2. This standard defines the DLNA mandatory and optional media format profiles for content.

5.4 Device categories

Device categories are a grouping of device classes that share common environmental characteristics (requirements) with system usages.

- In the interoperability guidelines, the following three device categories are defined: **Home network devices (HNDs)** are a group of device classes that share system usages in the home network with the same media format and network connectivity requirements.
- **Mobile handheld devices (MHDs)** are a group of device classes that share the same system usages as the HND device category, but have different requirements for media format and network connectivity.
- **Home infrastructure device (HID)** supports interoperability between device categories.

5.5 Device classes and roles

In the interoperability guidelines, the following five device classes are defined for the HND device category.

- A digital media server (DMS) with the role of exposing and distributing content.
- A digital media player (DMP) with the role of finding content exposed by a DMS and playing the content locally on the DMP.
- A digital media renderer (DMR) with the role of playing content it receives after being set up by another network entity.
- A digital media controller (DMC) with the role of finding content exposed by a DMS and matching it to the rendering capacities of a DMR and setting up the connections between the DMS and DMR.
- A digital media printer (DMP_r) with the role of printing images.

The following device classes are defined for MHD device category.

- A mobile digital media server (M-DMS) with the role of exposing and distributing content.
- A mobile digital media player (M-DMP) with the role of finding content exposed by an M-DMS and playing the content locally on the M-DMP.
- A mobile digital media uploader (M-DMU) with the role of sending content to an M-DMS with upload functionality.
- A mobile digital media downloader (M-DMD) with the role of finding and downloading content exposed by an M-DMS and playing the content locally on the M-DMD after downloading.
- A mobile digital media controller (M-DMC) with the role of finding content exposed by an M-DMS and matching it to the rendering capabilities of a DMR and setting up the connections between the server and renderer.

Many of these mobile device classes have counterparts in the HND device category; however, they differ from their counterpart at the network connectivity layer and at the media format layer in the DLNA architecture. For example, an M-DMC may be connected via mobile specific network connectivity while a DMC must meet the HND network connectivity requirements. This should not be taken to imply that MHD and HND devices cannot interact direct. In the discussion above and in the definition of terms of the MHD and HND device classes, mobile

device classes interact with other mobile device classes, such as an M-DMP interacting with an M-DMS. However, if the mobile and home devices have compatible network connectivity and can exchange compatible media format profiles, nothing in these statements should be taken to imply that an M-DMP cannot connect direct to a DMS to complete a system usage. See Table 2 and Table 3 for a listing of the required device interoperations and those that are only possible given compatible network and media format profile capabilities.

In order to allow HND and MHD devices to interact even with different network and media format profile capabilities, these guidelines define infrastructure devices of the HID device category that support interconnecting HND and MHD devices. The following device classes are defined within the HID device category.

- A mobile network connectivity function (M NCF) with the role of providing a bridging function between the MHD network connectivity and the HND network connectivity. An M NCF also provides support for security and power-saving modes. If a device in the MHD device category has only mobile specific network connectivity, it connects to home network via an M NCF. A device in the MHD device category with HND compatible network connectivity can connect to the home network direct.
- A media interoperability unit (MIU) with the role of providing content transformation between required media formats for the HND device category and the MHD device category. If a connected device in the MHD device category and a device in the HND device category have compatible media capabilities, they can interoperate direct without the need of an MIU. If they have different media capabilities, the MIU can be used to provide media interoperability.

The device functions that are incorporated in these device classes are illustrated in the figures that provide the details for system usages and their respective device interaction models in 5.7.2 through 5.7.8.

5.6 Device capabilities and roles

In these interoperability guidelines, the following device capabilities are defined.

- A push controller (+PU+) with the role of pushing its local content to a DMR.
- A printing controller-1 (+PR1+) with the role of controlling a DMP_r to print image content. A +PR1+ is also responsible for serving the image content and the XHTML-print document to the DMP_r.
- A printing controller-2 (+PR2+) with the role of controlling a DMP_r to print image content. Although a +PR2+ is not responsible for serving the image content, it is responsible for finding the image content on a DMS or M-DMS, providing the XHTML-print document to the DMP_r and instructing the DMP_r to print the images from the DMS or M-DMS.
- An upload controller (+UP+) with the role of sending content to a DMS or M-DMS with upload functionality.
- A download controller (+DN+) with the role of downloading content from a DMS or M-DMS to itself.

The device functions that are incorporated in these device capabilities are illustrated in the figures that provide the details for system usages and their respective device interaction models in 5.7.2 through 5.7.8.

5.7 System usages

5.7.1 Overview

In describing the flow of content in the system usages, indicated by the largest arrow in the system usage diagrams in 5.7.2 through 5.7.8, the terms push and pull are used. The terms push and pull are used in system usages to characterize the user's perception of the source or sink location in the process of content transfer. That is, "pull" means that the content is travelling to the user, while "push" means that the content is travelling from the user. This

perception is not a reflection of the technical underlying transport mechanism utilized to perform the transfer the content from the source to the sink (for example, HTTP and RTP).

In the interoperability guidelines, the following seven system usages are defined that map to all of the use case scenarios being enabled by the detailed guidelines.

- 2-box pull system usage

This usage involves a user at a DMP or an M-DMP, which enables the user to find and play content that is advertised and distributed by a DMS or M-DMS.

- 2-box push system usage

This usage involves a user at a push controller, which enables the user to distribute content to a DMR for playback purposes.

- 3-box system usage

This usage involves a user at a DMC or an M-DMC, which enables the user to find content on a DMS that in turn will be played on a user-selected DMR.

- 2-box printing system usage

This usage involves a user at a printing controller-1, which enables the user to set up image print tasks with a DMP.

- 3-box printing system usage

This usage involves a user at a printing controller-2, which enables the user to find images on a DMS or M-DMS and then set up a print tasks with a DMP.

- Download system usage

This usage involves a user at a download controller or an M-DMD, which enables the user to download content from a DMS or an M-DMS so that the download controller or the M-DMD has its own copy.

- Upload system usage

This usage involves a user at an upload controller or an M-DMU, which enables the user to send content to a DMS or an M-DMS with the upload device option so that the DMS or the M-DMS can distribute the content to other endpoints.

Subclause 5.7.2 through 5.7.8 will briefly describe each of the system usages and their respective device interaction models. In these system usages, devices in the MHD and HND device categories can achieve interoperability by utilizing an MIU with the responsibility for media format level conversions, and/or an M-NCF for bridging the networking connectivity layers between the two device categories. These elements are described in 5.8. For clarity, the device interaction model diagrams in 5.7.2 through 5.7.8 show HND and MHD devices interacting direct. All these system usages are shown with device functions that are media transport agnostic. All of these system usages imply HTTP as the mandatory media transport and other optional media transports, such as RTP, can be substituted.

5.7.2 2-box pull system usage

The 2-box pull system usage pulls DLNA compliant content from a media server (DMS/M-DMS) to be rendered locally by the device pulling the content (DMP, M-DMP). The user perspective is that content is being pulled to the DMP or the M-DMP for immediate rendering on the device. The user is browsing and selecting content on the DMS or the M-DMS. Note that the rendering function is not exposed onto the network in a DMP or an M-DMP implementation. Also note that in all of the following system usage diagrams, the media transport client/server is for the media transport layer only. The UPnP Device/CP has HTTP functions independent of the media transport layer and is implied as being part of the UPnP device/CP device functions. Figure 3 illustrates this device interaction model. The following steps are performed in this system usage.

- Invoke UPnP actions to browse and select content.
- Request the content for playback.

c) Transport the content to the DMP or the M-DMP.

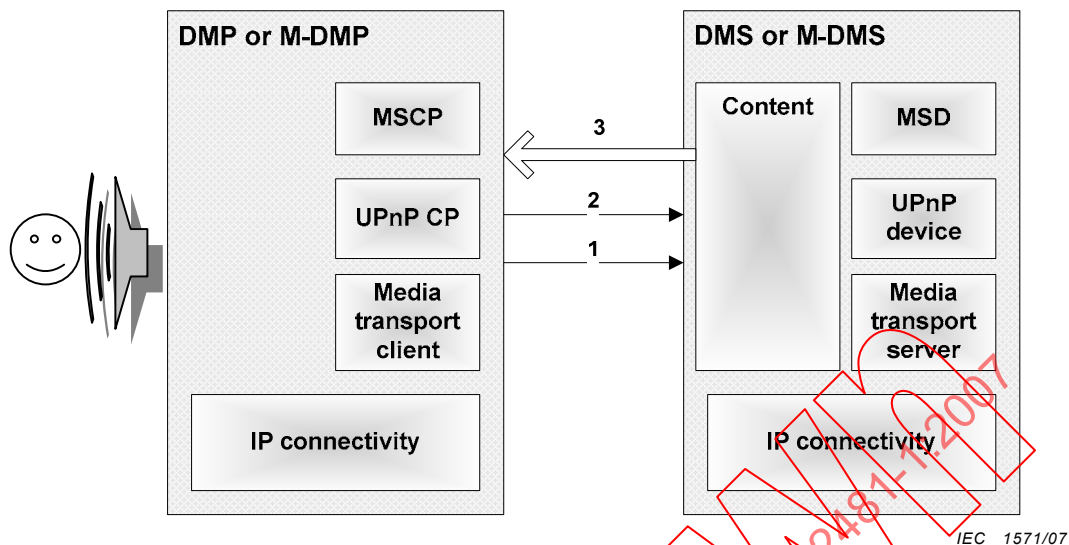


Figure 3 – 2-box pull system usage interaction model

5.7.3 2-box push system usage

The 2-box push system usage pushes DLNA compliant content to a rendering device (DMR). The user perspective is that content is being pushed to the DMR even though content may actually be transported in a "pull" manner depending on the media transport used. The user is selecting content at the device where the content is resident. Figure 4 illustrates this device interaction model. The following steps are performed in this system usage.

- Invoke UPnP actions to set up a playback session.
- Request the content for playback.
- Transport the content to the DMR.

Note that the push controller device capability functionality can only be incorporated as part of any physical device with a valid DLNA device class. It must never appear as a stand-alone device. This is how the push controller device capability inherits other device functions (for example, IP connectivity) at other layers in the DLNA device architecture. This is applicable to device classes in both the HND and MHD device categories.

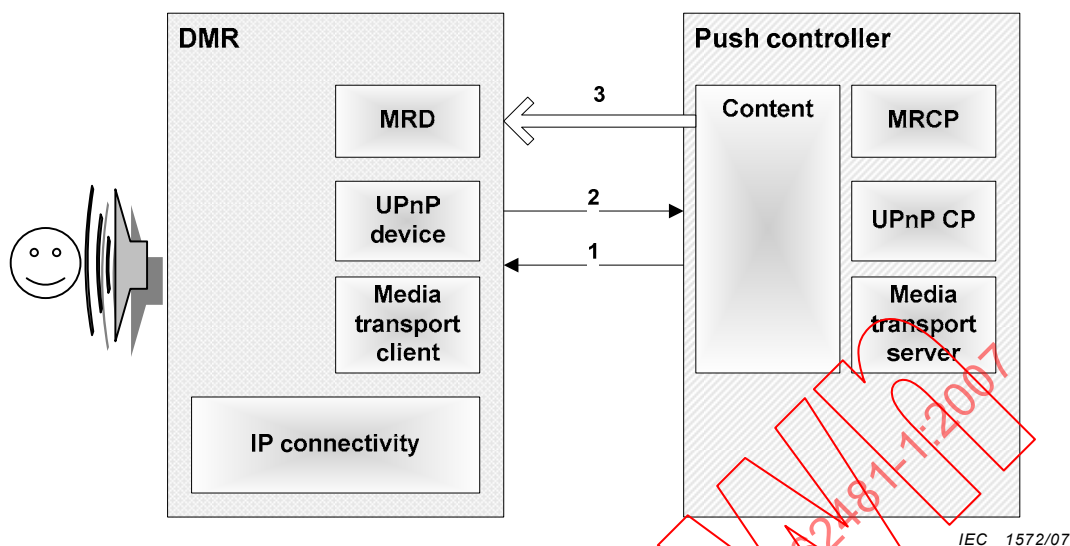


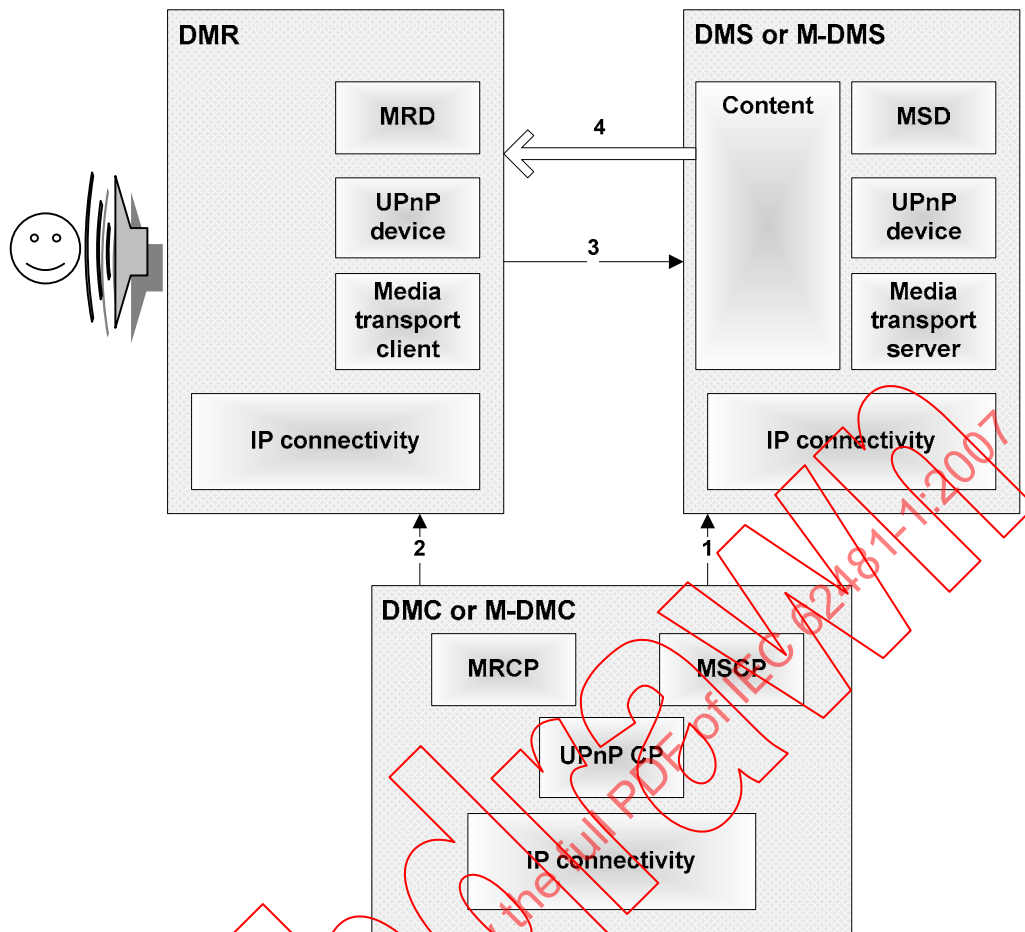
Figure 4 – 2-box push system usage interaction model

5.7.4 3-box system usage

The 3-box system usage uses a device controller (DMC/M-DMC) to browse content on a media server (DMS/M-DMS) and to select a rendering device (DMR) to play the selected content. The DMC or the M-DMC is responsible for making sure a DMR can render the selected DLNA content.

Figure 5 illustrates this device interaction model. The following steps are performed in this system usage.

- a) Invoke UPnP actions to browse and select content.
- b) Invoke UPnP actions to verify that the DMR has the capability to render the selected content and then set up a connection for the selected content between the DMR and the DMS or the M-DMS.
- c) Request the content for playback.
- d) Transport the content to the DMR.



IEC 1573/07

Figure 5 – 3-box system usage interaction model

5.7.5 2-box printing system usage

The 2-box printing system usage has a printing controller-1 that instructs an image printing device (DMP) to print image(s) served locally by the printing controller-1. Print jobs are XHTML-print documents that are sourced by the printing controller-1 and all images referenced in an XHTML-print document are served up locally by the printing controller-1.

Figure 6 illustrates this device interaction model. The following steps are performed in this system usage:

- Invoke UPnP actions to the DMP to initiate the print job by providing the URL to the XHTML-print document.
- The DMP acquires the XHTML-print document from the printing controller-1.
- The DMP requests the image content from the printing controller-1.
- The printing controller-1 responds with the image content.

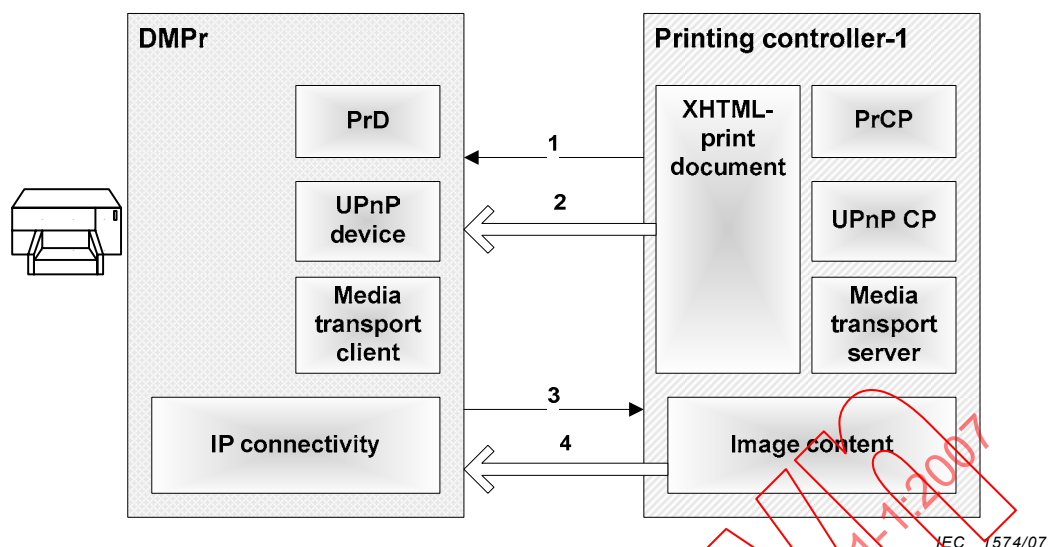


Figure 6 – 2-box printing system usage interaction model

5.7.6 3-box printing system usage

The 3-box printing system usage has a printing controller-2 that instructs an image printing device (DMPr) to print image(s) served by a DMS or M-DMS device class. The printing controller-2 has the additional functionality for finding and selecting the image(s) exposed by a DMS and/or M-DMS. Print jobs are XHTML-print documents that are sourced by the printing controller-2 and all images referenced in an XHTML-print document are served up by a DMS and/or M-DMS.

Figure 7 illustrates this device interaction model. The following steps are performed in this system usage.

- Invoke UPnP actions to browse and select image content.
- Get the URLs for the selected image content and reference the URLs in the XHTML-print document.
- Invoke UPnP actions on a DMPr to initiate the print job by providing the URL to the XHTML-print document.
- The DMPr acquires the XHTML-print document from the printing controller-2.
- The DMPr requests the Image content from the DMS or M-DMS.
- The DMS or M-DMS responds with the image content.

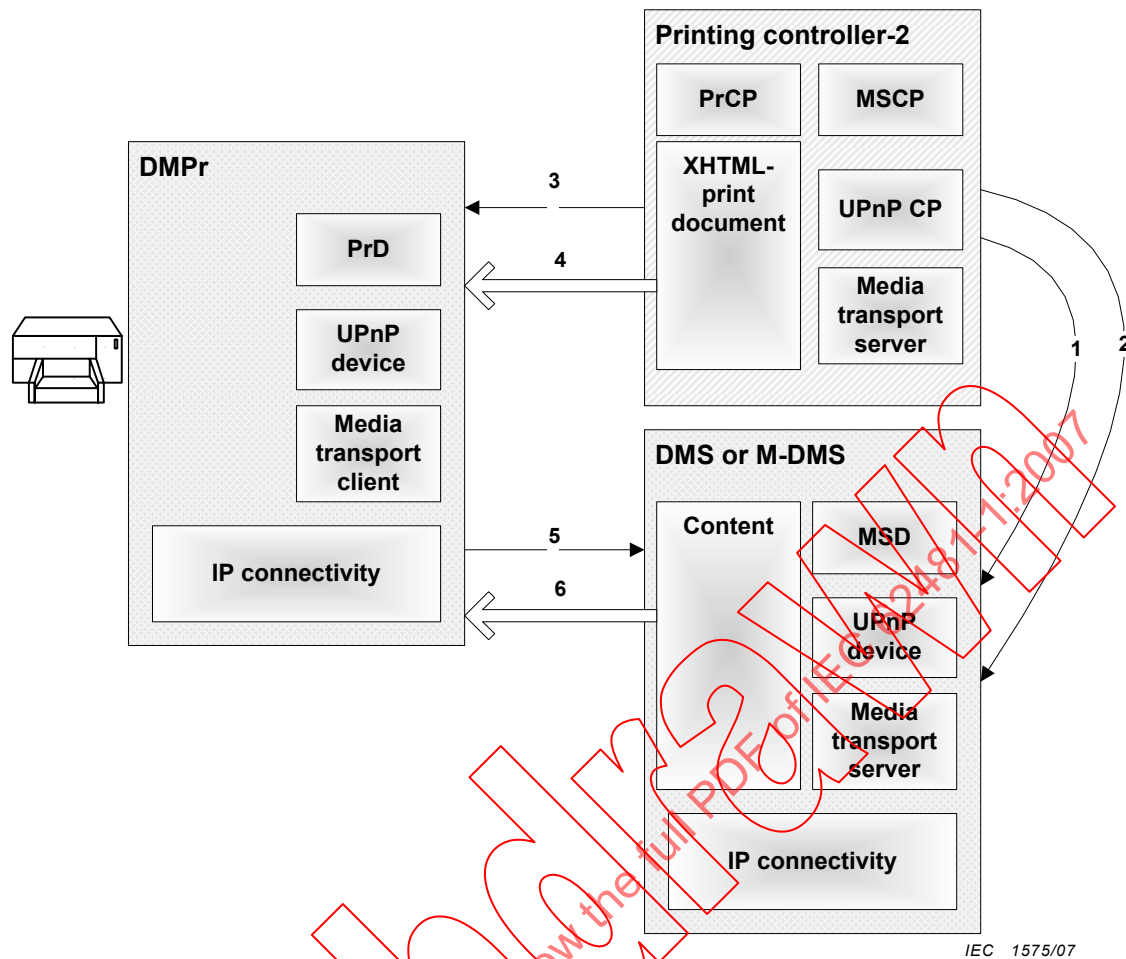


Figure 7 – 3-box printing system usage interaction model

5.7.7 Download system usage

The download system usage allows a download controller or an M-DMD to transfer and store DLNA content from a media server (DMS or M-DMS).

Figure 8 illustrates this device interaction model. The following steps are performed in this system usage.

- Invoke UPnP actions to find content to download.
- Request the content that needs to be downloaded.
- Transport content to the download controller or the M-DMD.

Note that the download controller device capability functionality can only be incorporated as part of any physical device with a valid DLNA device class. It can never appear as a stand-alone device. This is how the download controller device capability inherits other device functions (for example, IP connectivity) at other layers in the DLNA device architecture. In the MHD environment, this system usage can be accomplished by a device with only this functionality, hence the need for an M-DMD device class to provide support for all layers in the DLNA architecture. This is not a requirement in the HND environment which must incorporate this functionality as an addition to an existing device class.

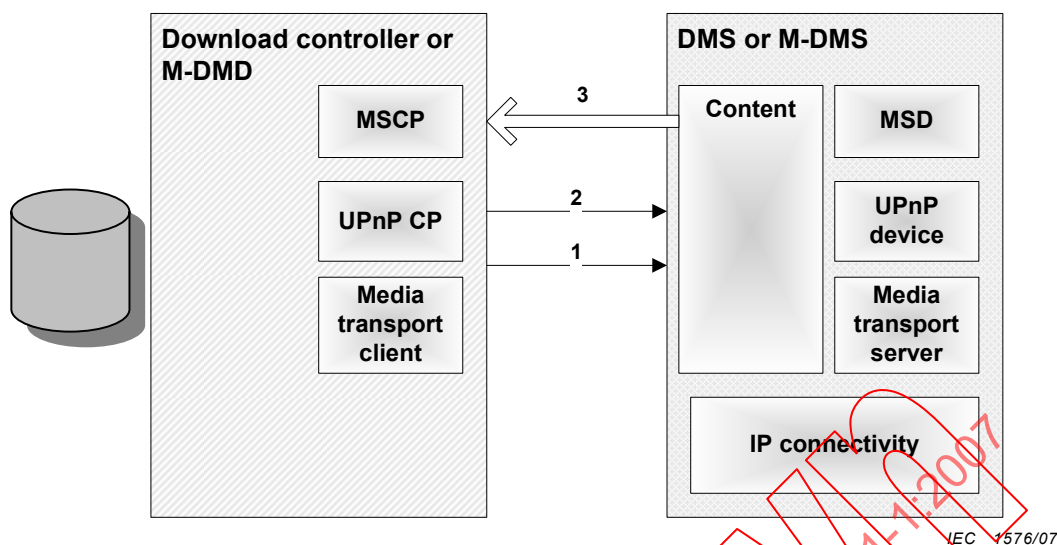


Figure 8 – Download system usage interaction model

5.7.8 Upload system usage

The upload system usage has an upload controller device capability or an M-DMU to instruct a media server (DMS/M-DMS) to accept some new content to be added to its list of available content.

Figure 9 illustrates this device interaction model. The following steps are performed in this system usage.

- Invoke UPnP actions to create a CDS entry for the content to be uploaded.
- Transport the content being uploaded to the DMS or the M-DMS.

Note that the upload controller device capability functionality can only be incorporated as part of any physical device with a valid DLNA device class. It must never appear as a stand-alone device. This is how the upload controller device capability inherits other device functions (for example, IP connectivity) at other layers in the DLNA device architecture. In the MHD environment, this system usage can be accomplished by a device with only this functionality, hence the need for an M-DMU device class to provide support for all layers in the DLNA architecture. This is not a requirement in the HND environment which must incorporate this functionality as an addition to an existing device class.

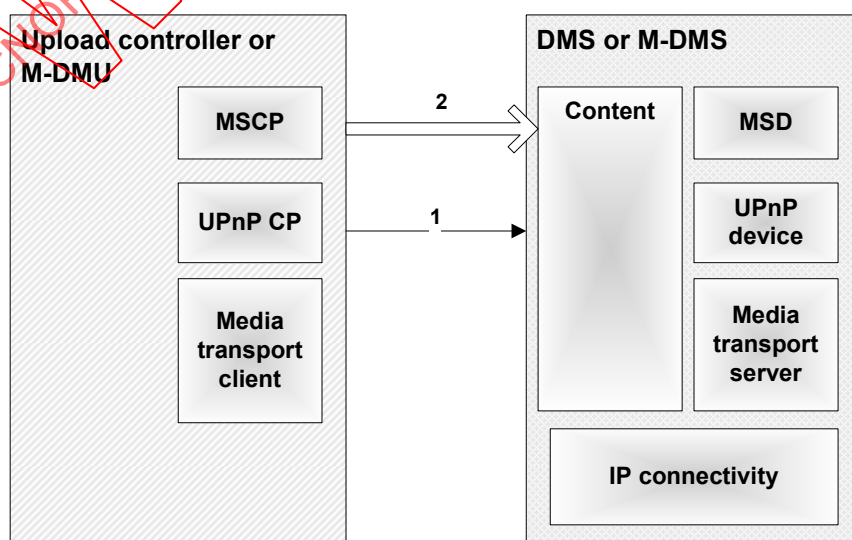


Figure 9 – Upload system usage interaction model

5.8 Home infrastructure device (HID) system usage

5.8.1 General

As described in 5.1, the HID device category contains device classes that enable interoperability for common system usages between different device categories. In particular, the HND and MHD device categories share similar system usages, but they differ in their network connectivity and media format requirements. The device class for bridging the network connectivity layer is the M-NCF device class. The device class for bridging the media format layer is the MIU device class. Both of these entities are described in 5.8.

Figure 10 illustrates these bridging functions for the 2-box pull system usage between device classes in the MHD and HND device categories. The same concept can be applied to all of the system usages described previously. In summary, network layer traffic is bridged by the M-NCF for communications between the MHD and HND device categories as needed, and the MIU provides content transformation between the HND and MHD device categories as needed.

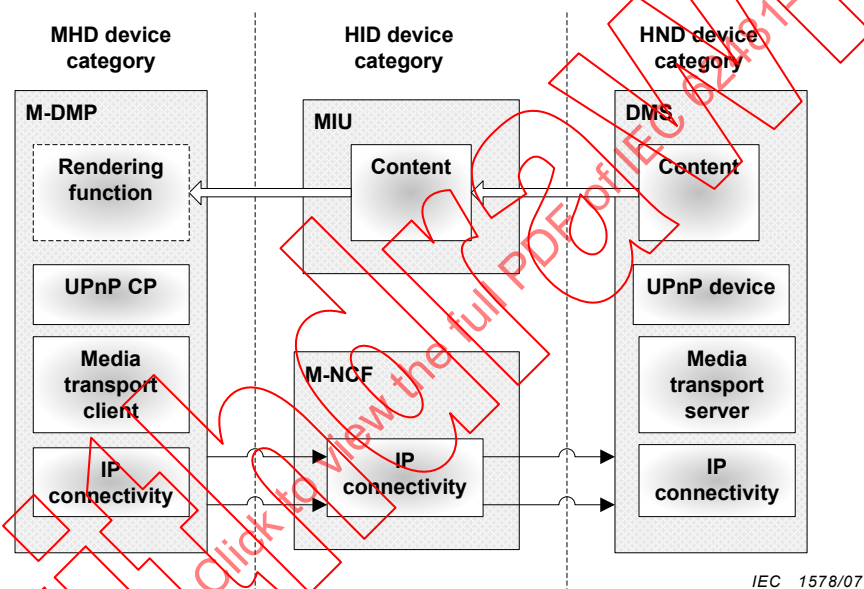
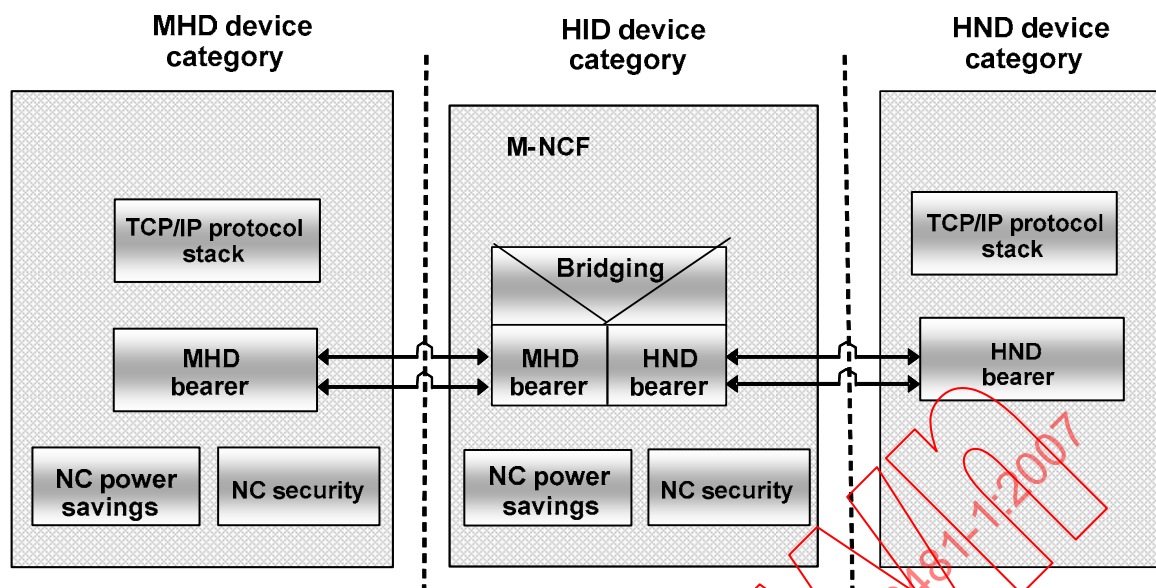


Figure 10 – 2-box pull system usage interaction model between device categories

5.8.2 Bridging HND and MHD network connectivity

Because devices in the HND and MHD device categories have different form factors, energy requirements and usages, they may support different link-level bearers to provide connectivity. For instance, it is common for MHD devices to support short-range and low-power consumption wireless bearers, such as Bluetooth. Furthermore, besides selection of bearers, MHD devices usually have different requirements from HND devices in two other areas at the network connectivity (NC) level, namely support for NC power savings and NC security. For instance, different requirements in the area of NC power savings may be due to the fact that most MHD devices are battery-operated, while in the area of NC security to the fact that MHD devices may belong to visitors who only need to be given temporary access to the home network. To bridge the network connectivity gap between MHD and HND device categories that is created by possibly different bearers and different needs in NC power savings and NC security, the mobile network connectivity function (M-NCF) device class is introduced. An example of the operation of the M-NCF is illustrated in Figure 11.



IEC 1579/07

Figure 11 – M-NCF bridging the network connectivity gap between MHD and HND device categories

Devices in the MHD device category may support different bearers from those supported by devices in the HND category. It is the role of the M-NCF to do link-level bridging between different MHD and HND bearers. This link-level bridging function is transparent at the IP connectivity level and above. Furthermore, the M-NCF has the role of providing support for NC power savings to MHD devices that wish to conserve energy. This is done by leveraging the underlying power saving mechanisms provided by the MHD bearers and by, optionally, performing traffic reduction operations to prevent excess multicast and broadcast traffic from reaching the MHD devices. Finally, it is the role of the M-NCF to provide authenticated and encrypted access to MHD devices to the home network, whenever desired by its owner-administrator. To achieve that, the M-NCF provides required functionality to establish, manage and revoke the necessary permanent or temporary access rights for any MHD devices to be able to connect to the home network.

It is important to note that not all MHD devices are required to connect to the home network via an M-NCF. For example, an MHD device which supports a bearer which is also an HND bearer may connect to the home network without an M-NCF, in the same way that any HND device supporting this bearer would. However, whenever an M-NCF supporting the necessary MHD bearer exists, it is recommended that MHD devices connect to the home network via the M-NCF, to ensure better interoperability and better meeting the unique requirements of the MHD devices.

5.8.3 Bridging HND and MHD media formats

Because devices in the HND and MHD device categories have different form factors and usages, they also differ in the mandatory media format profiles which are supported. For instance, all HND devices that operate on audio/video data must be able to process a profile based on MPEG2, while all MHD devices that operate on audio/video data must be able to process a profile based on MPEG 4 Part 10. This leads to a potential lack of interoperability between devices of the different categories. The media interoperability unit (MIU) is a device class of the HID device category that is defined to bridge this gap for the mandatory media format profiles and ensure media level interoperability between devices of the MHD and HND device categories. The operation of the MIU device class is shown in the following diagram.

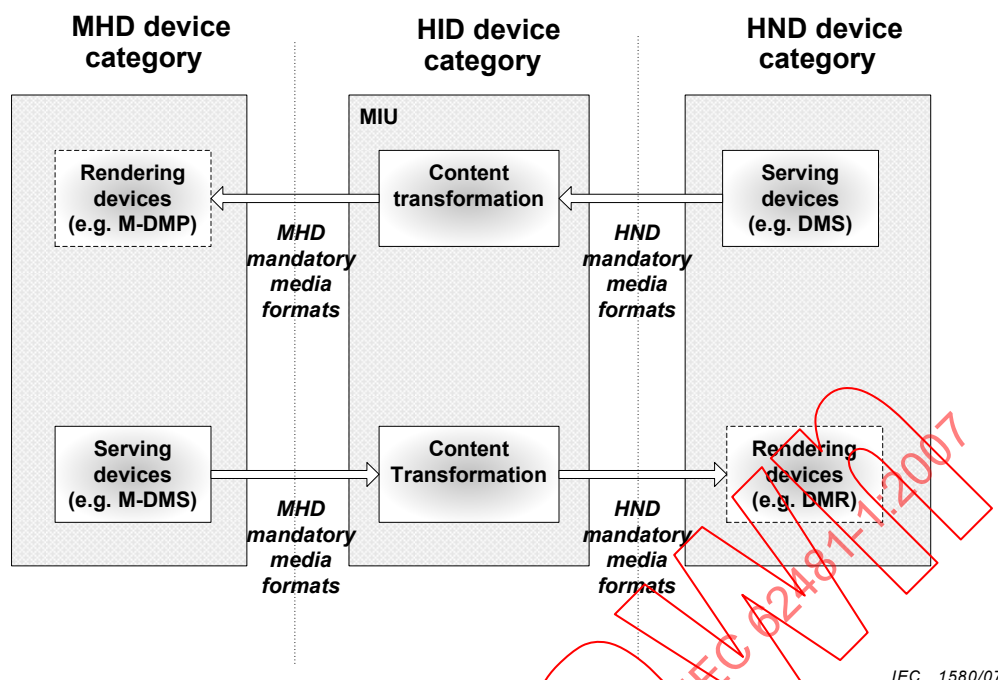


Figure 12 – Media interoperability between device categories

The devices of the MHD or HND device categories can serve or consume content in the mandatory media format profiles of the given device category. It is the role of the MIU to convert between the media format profiles so that content created and stored on an MHD device can be rendered on an HND device and vice versa.

It is important to note that the MIU is responsible only for media format level conversions. Because the core architecture of both the MHD and HND devices is the same set of UPnP devices, MHD devices are free to connect direct with their peer HND devices at any time, provided that they abide by the HND media format requirements. For example, MHD servers may find a DMR that can render their particular media format profiles, but they must not expect that functionality from an HND device. In the case where the HND device does not meet the media requirements direct, the MHD device should connect to the MIU for the necessary services.

5.9 Interoperability guidelines usage

The guideline requirements tables found in Clause 7 contain a column that specifies which device classes apply to a requirement. For the interoperability guidelines, five device classes are defined for the HND device category. They are a DMS, DMP, DMC, DMR, and DMP. The MHD device category with five device classes is introduced along with the two device classes of the HID device category. Table 2 summarizes all of the device classes in the HND device category and the mnemonics used within the interoperability guidelines. Table 3 summarizes all of the device capabilities that can be deployed with any device class and the mnemonics used within the interoperability guidelines. Table 4 summarizes all of the device classes in the MHD device category and the mnemonics used for these device classes. Table 5 contains the device classes in the HID device categories and the mnemonics used for these device classes.

Table 2 – DLNA device classes in the HND device category

DLNA device class	Media management components	Media transport components	Functional description	Device classes or capabilities interacted with for defined system usages	Device classes interacted with given compatible networking and media formats profiles
DMS	MSD	Media	Serves up media	DMP, DMC, DMR,	M-DMP, M-DMC,

(digital media server)		transport server		DMP, other endpoints with +UP+, +DN+, or +PR2+ capabilities	M-DMD, M-DMU
DMP (digital media player)	MSCP	Media transport client	Selects, controls and renders the selected media	DMS	M-DMS
DMC (digital media controller)	MSCP MRCP	N/A	Controls the content selection and content rendering between networked devices	DMS, DMR	M-DMS
DMR (digital media renderer)	MRD	Media transport client	Renders content	DMC, DMS, other endpoints with +PU+ capabilities	M-DMC, M-DMS
DMP (digital media printer)	PrD	Media transport client	Prints images	DMS, other endpoints with +PR1+ or +PR2+ capabilities	M-DMS

A device capability can be applied to any valid DLNA device class. Table 3 summarizes all of the device capabilities used in the system usages and the mnemonics used within these interoperability guidelines to specify which requirements apply to them.

Table 3 – DLNA device capabilities

DLNA device capability	Device capability controller identifier	Applicable device classes	Media management components	Media transport components	Required device classes interacted with for defined system usages	Device classes interacted with given compatible networking and media formats profiles
Push controller	+PU+	Any	MRCP	Media transport server	DMR	N/A
Printing controller-1	+PR1+	Any	PrCP	Media transport server	DMP	N/A
Printing controller-2	+PR2+	Any	PrCP MSCP	Media transport server	DMP, DMS	M-DMS
Download controller	+DN+	Any	MSCP	Media transport client	DMS	M-DMS
Upload controller	+UP+	Any	MSCP	Media transport client	DMS	M-DMS

The MHD device category has different media format and network connectivity requirements because of various device constraints. Table 4 summarizes all of the device classes in the MHD device category and the mnemonics used within the interoperability guidelines.

Table 4 – DLNA device classes in the MHD device category

DLNA device class	Media management components	Media transport components	Functional description	Device classes interacted with for defined	Device classes or capabilities interacted with given

				system usages	compatible networking and media formats profiles
M-DMS (mobile digital media server)	MSD	Media transport server	Serves up media	M-DMP, M-DMC, M-DMD, M-DMU	DMP, DMC, DMR, DMPr, other endpoints with +UP+, +DN+, or +PR2+ capabilities
M-DMP (mobile digital media player)	MSCP	Media transport client	Selects, controls and renders the selected media	M-DMS	DMS
M-DMC (mobile digital media controller)	MSCP MRCP	N/A	Controls the content selection and content rendering between networked devices	M-DMS, DMR	DMS
M-DMU (mobile digital media uploader)	MSCP	Media transport client	Uploads the selected media to servers	M-DMS	DMS
M-DMD (mobile digital media downloader)	MSCP	Media transport client	Selects, controls and downloads the selected media	M-DMS	DMS

Due to the differences in the media format profile support and network connectivity requirements, the interoperability for device classes is only assured within a device category. To extend interoperability for the device classes in the MHD device category throughout the home network, including the device classes in the HND device category, a special device category named home infrastructure device (HID) is defined. The HID device category ensures interoperability between devices of different categories and ensures interoperability between the device classes and capabilities and the devices listed in the final two columns in the above tables. Table 5 summarizes all of the device classes in the HID device category and the mnemonics used within the interoperability guidelines.

Table 5 – DLNA device classes in the HID device category

DLNA device class	Media management components	Media transport components	Functional description
M-NCF (mobile network connectivity function)	N/A	N/A	Provides a network connectivity bridge between devices in the HND and MHD device categories
MIU (media interoperability unit)	MSD, MRD, MSCP, MRCP	Media transport server Media transport client	Provides virtual services for content transformation between required media formats for devices in the HND and MHD device categories

6 Guideline terminology and conventions

6.1 Guideline compliance classifiers

IETF RFC 2119 provides a description of terminology conventions used in all IETF RFC documents. The terminology and conventions used by the interoperability guidelines are adapted from this reference. The details of each guideline will carry a compliance classifier from the following set.

[M]ust, Required, Must:

This is the minimum set of requirements that will ensure interoperability and/or robust operation between devices. All devices are expected to comply with these requirements when expressed in unconditional form. A conditional requirement expressed in the form, "If X, then Y must be implemented", means that the requirement "Y" must be met when the conditional aspect "X" applies to a given implementation.

- [S]hould, Recommended:** Recommended items are optional items that are strongly recommended for inclusion in products. The difference between "recommended" items and "optional" items, below, is one of priority. When considering features for inclusion in a product, recommended items should be included first.
- [O]ptional, May:** Optional items are suggestions for features that will enhance the user experience or are offered as a less preferred choice relative to another recommended feature. If optional features are included, they must comply with the requirement to ensure interoperability with other implementations.

6.2 Standard or specification usage classifiers

When specifying guideline details, it is often useful to reiterate or clarify certain aspects of a standard or specification that are often violated or misunderstood. Furthermore, there may be guideline requirements that intentionally contradict or restrict implementation of certain aspects of a standard or specification in order to ensure interoperability between DLNA devices. The following classifiers are used in the interoperability guidelines to indicate the relationship of a specific guideline requirement to a source standard or specification.

- [A]dding:** A guideline requirement that adds to or supplements a standard or specification to enhance interoperability. A guideline requirement that does not reference a standard or specification also uses this classifier.
- [C]larifying:** A guideline requirement that addresses vague or ambiguous aspects of a standard or specification.
- [F]ixing:** A guideline requirement that intentionally supersedes and fixes aspects of a standard or specification that is incorrect and would otherwise provide a poor user experience or prevent device interoperability.
- [L]imiting:** A guideline requirement that narrows or specifies an exact behaviour in areas where a standard or specification provides for greater degrees of latitude in implementation.
- [R]epeating:** A guideline requirement that repeats what is already in a standard or specification because of observed and repeated problems with implementations. Whenever a guideline requirement with this usage classifier seems to be in conflict with the actual standard, the standard prevails over the guideline requirement.

6.3 Guideline font usage conventions

The following font usage conventions are used within the interoperability guidelines to provide additional clarity.

- Hyperlinks to reference citations are indicated as [number]. For example [1]¹, [20], ...
- UPnP action names are indicated as: [Service acronym]:[action name], such as CDS:Browse.

6.4 Guideline syntax notation conventions

The following are syntax (BNF) notation conventions used within the interoperability guidelines to provide readability.

- Linear whitespace (LWS) characters, such as carriage returns, spaces, tabs, or line feeds, are not implied anywhere in any of the syntax (BNF) definitions used within the interoperability guidelines.

¹ Figures in square brackets refer to the Bibliography.

- The use of LWS characters is restricted within the interoperability guideline unless explicitly specified in any of the syntax definitions with reference to UPnP HTTP communications.
- By default, text tokens and values have a case-sensitive treatment unless explicitly noted in the guidelines. This convention also applies to BNF definitions, XML tag names, XML tag values, capability IDs, and HTTP header values for HTTP headers used by the DLNA guidelines. One of the exceptions to this rule applies to the names of HTTP headers. HTTP header names have a case-insensitive treatment. For example, TimeSeekRange.dlna.org is the same as timeseekrange.dlna.org (see 7.5.5.2.6). Other exceptions are described in each guideline which defines BNF syntax.

6.5 Guideline normative and informative text conventions

All text that appears in the interoperability guidelines is to be considered normative unless explicitly stated otherwise, such as informative references and informative annexes. Normative text includes introductory text before guideline requirement tables, but testable requirements are only contained within guideline requirement table entries.

6.6 DLNA XML namespaces and schemas

The interoperability guidelines make numerous references to XML elements and attributes that are defined for DLNA device classes and device capabilities. However, these namespaces are intentionally not defined through a formal DLNA XML schema. This allows the interoperability guidelines to define new XML elements and attributes in the future, without having to define a new namespace or schema definition. DLNA devices classes and device capabilities are expected to exhibit tolerant behaviour when encountering XML elements or attributes that are defined in the future, as required by existing guidelines 7.3.24 and 7.4.1.3.1. The following table lists the namespace values that are used by the interoperability guidelines and the context for their usage.

Table 6 – DLNA namespace values

DLNA namespace values	
Namespace value	Usage context
urn:schemas-dlna-org:device-1-0	Used for XML elements and attributes defined by DLNA interoperability guidelines for use in UPnP device description files
urn:schemas-dlna-org:metadata-1-0/	Used for XML elements and attributes defined by DLNA interoperability guidelines for use in DIDL-Lite documents and fragments

7 Guideline requirements

7.1 General

This clause covers the guidelines that enable vendors to build interoperable products. Devices built to the interoperability guidelines will be able to manage, transfer, and play personal media over a home network.

These guidelines are in a clause/subclause format as shown in Figure 13.

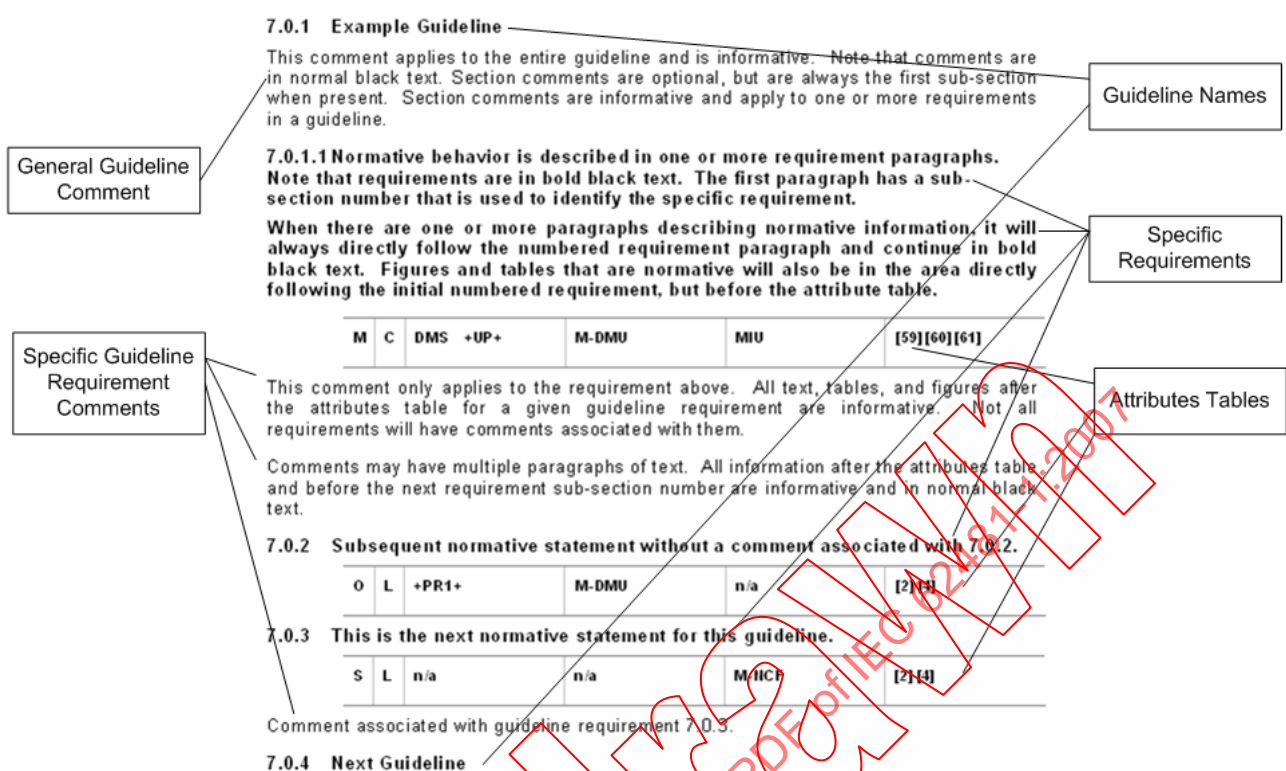


Figure 13 – Guideline layout and definitions

The following list describes the content of Figure 13.

- Name:** A unique label for the guideline. The label is preceded with a sequentially increasing number to allow easy lookup.
- Requirements:** The actual description of a guideline. A guideline is preceded with a sequentially increasing number to allow easy lookup. A given guideline may consist of several sub-requirements that are also numbered.
- Attribute table:** A summary of the essential attributes of a requirement. The table is a single row with the following definitions for the columns.
 - Compliance classifier: M/S/O (see 6.1 for the definition of guideline compliance classifiers).
 - The specification usage classifier: A/C/F/L/R: for the guideline (see 6.2 for the definition of specification usage classifiers).
 - HND device classes with device capabilities (see Table 2 and Table 3 for definitions). Device capabilities were originally defined within the context of the HND device category and are listed in the HND column of the attribute table. Device capabilities can also apply equally to the MHD device category but have been omitted from the MHD column in the attribute table to provide for better readability.
 - MHD device classes (see Table 4 for definitions).
 - HID device classes (see Table 5 for definitions).
 - Ref No.: Standards that are referenced by the guideline. Standards citations are by number and are defined in Clause 2 and in the Bibliography.

Guideline attribute columns that do not have a value have the designation "n/a" (not applicable). A visual map of possible values for the attribute tables is given in Figure 14.
- Comment:** Supplementary information about a guideline such as a justification for the guideline, the specific interoperability issue that is addressed, etc.

Note that many guidelines do not explicitly list MIU since guidelines which apply to a device class also apply to the virtualized variants.

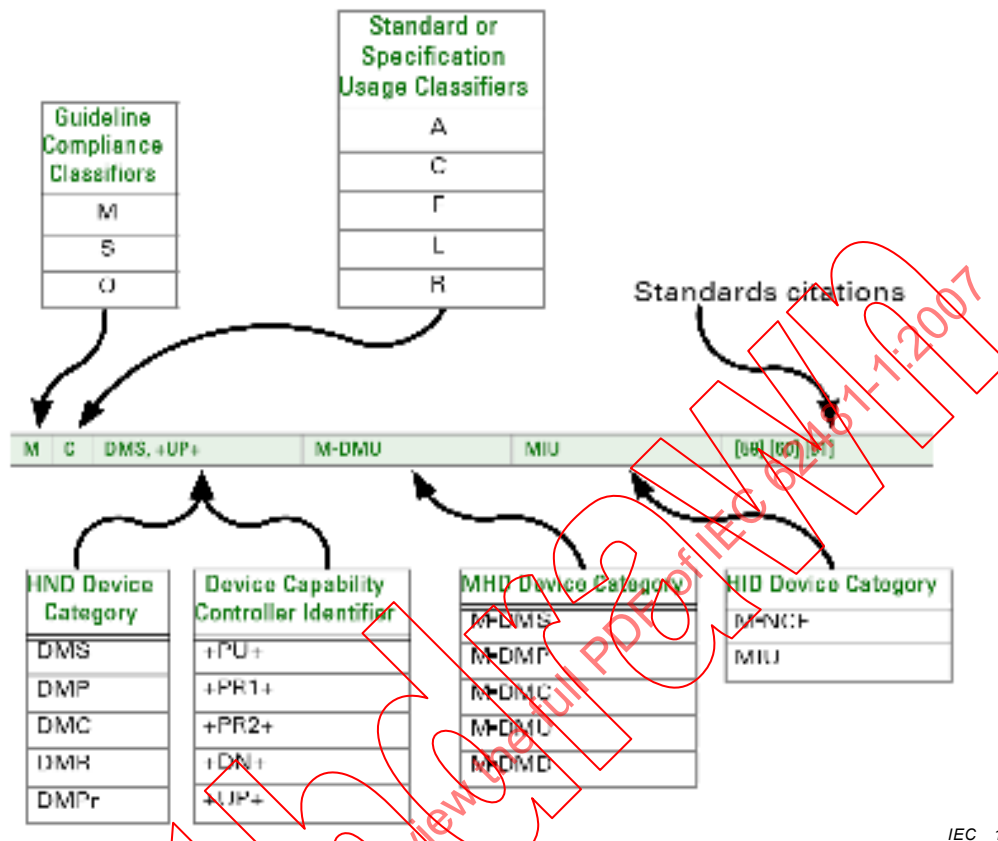


Figure 14 – Visual map of possible values for the attribute tables

- Conditions for measuring time in message exchanges

These guidelines define in certain cases time constraints for the exchange of messages between two communicating endpoints. These time constraints have been defined as a means to provide some operational consistency between the two communicating endpoints. However, in best-effort networks, actual time measurements for exchanging messages show wide variations depending on perturbations derived from network conditions, traffic, available bandwidth, and others. For this reason, this clause includes recommendations for conditions at the time of making these measurements.

- The two communicating endpoints should establish communications under ideal network conditions.
- Time measurements at a given layer assume that the underlying layers preserve the communication channel. For example, time measurements at the HTTP layer cannot be valid if the underlying TCP/IP channel breaks during the measurements.
- Unless specified otherwise, time measurements assume that the communicating devices are both in active mode. This means that time measurements should not include transitions from sleep mode to active mode.

7.2 Networking and connectivity

7.2.1 General

Networking and connectivity between devices is fundamental to the interoperability guidelines. The family of protocols known as the Internet protocol (IP) is the backbone for home network connectivity. Clusters of devices in the home may use other interconnect technologies, but IP

ties these clusters together within the home, and provides connectivity outside the home to the global Internet. IP is independent of physical media, and, therefore, there are a variety of connectivity options for DLNA devices.

The networking and connectivity guidelines are organized in the following subclauses.

7.2.2 Normative definitions of NC-PS modes

7.2.3 Networking and connectivity – General capability requirements

7.2.4 Networking and Connectivity – QoS requirements

7.2.5 Networking and connectivity – Device

7.2.2 Normative definitions of NC-PS modes

Mobile handheld devices are typically battery-powered, and, therefore, saving power is a very important topic. Typically, these devices utilize power-saving mechanisms both internally (for example, place the processor in a low power state, turn off the screen, etc.) and at the connectivity level to reduce their power utilization and extend their battery life. The NC-PS modes refer to power savings on the connectivity link between a mobile handheld device and an M-NCF. Internal power saving measures are beyond the scope of the NC-PS guidelines. The NC-PS mode definitions are shown in Table 7. The definitions in Table 7 are normative because normative DLNA guidelines make reference to these NC-PS modes. The NC-PS modes are only applicable for NC Bluetooth.

Table 7 – Normative definitions of network connectivity power-saving (NC-PS) modes

NC-PS mode	Definition
Active	The connection between a mobile handheld device and an M-NCF is in the 'active' mode when the link-level connection has been established and no power saving mechanism is used in the underlying bearer used to transfer TCP/IP protocol-stack traffic. The mobile handheld device has full IP connectivity at the highest possible data-rate and least latency
Standby	The connection between a mobile handheld device and an M-NCF is in the 'standby' mode when the link-level connection has been established and both the mobile handheld device and the M-NCF have collaborated to put the link of the underlying bearer in a lower power state, using the mechanisms supported by that bearer. The mobile handheld device still has full IP connectivity, but data-rates and latency may be affected adversely
Disconnected	The connection between a mobile handheld device and an M-NCF is in the 'disconnected' mode, when the link-level connection of the underlying bearer is disconnected. The mobile handheld device has no IP connectivity

7.2.3 Networking and connectivity – General capability requirements

7.2.3.1 General

The guidelines in this clause provide requirements for general capabilities. For example, these requirements describe the baseline capabilities of any Ethernet or Bluetooth implementation.

7.2.3.2 General capability requirements for Ethernet

7.2.3.2.1 NC Ethernet – Base

If Ethernet is supported, IEEE 802.3i (10BASE-T) and IEEE 802.3u (100BASE TX) with auto negotiation capability and a connection to the network provided by an RJ45 connector is required.

M	R	n/a	n/a	n/a	IEEE 802.3

7.2.3.2.2 NC Ethernet – Cabling

If Ethernet is supported, any supplied network cabling should have a rating of Category 5e or better.

S	R	n/a	n/a	n/a	[12]

7.2.3.2.3 NC Ethernet – Gigabit

If Ethernet is supported, IEEE 802.3ab (1000BASE T) is recommended in addition to 7.2.3.2.1. An implementation must support auto negotiation of gigabit operation with a similarly capable link partner and drop down to a lower speed as appropriate.

S	R	n/a	n/a	n/a	IEEE 802.3

Gigabit Ethernet is becoming available and affordable for home networks.

7.2.3.2.4 NC Ethernet – QoS tolerance

If Ethernet is supported, incoming tagged packets must be tolerated. Tagged packets are Ethernet packets that include priority tags conformant with IEEE 802.3, 3.5. Here, “tolerate” means that the packet payload of any received tagged or untagged packet must be properly passed up to the higher layers in the network stack.

M	R	n/a	n/a	n/a	IEEE 802.3

Packet tagging is the only QoS mechanism available on Ethernet at the link layer. Many devices on home networks are already capable of sending tagged frames, so all devices must be able to tolerate them. For guidelines on tagging, see NC Ethernet DLNAQOS – Conformance.

7.2.3.2.5 NC 802.11 – Base

7.2.3.2.5.1 If 802.11 is supported, one or more of the following radio selections is allowed:

- 802.11a;
- 802.11b;
- 802.11g.

For example, 802.11a, 802.11b, 802.11g, 802.11a/b, 802.11b/g, and 802.11a/b/g all meet this requirement.

M	R	n/a	n/a	n/a	IEEE 802.11a IEEE 802.11b IEEE 802.11g

There is no implied requirement that a device needs to support multiple radios nor is it prohibited.

See Annex A for recommendations on wireless access points and how they will help enable interoperability between products with different radio selections.

7.2.3.2.5.2 If 802.11 is supported, the implementation must support infrastructure mode operation.

M	R	n/a	n/a	n/a	IEEE 802.11a IEEE 802.11b IEEE 802.11g

Some DLNA device classes may be required to support *ad hoc* (IBSS) mode for Wi-Fi conformance; however, the interoperability guidelines do not provide any requirements for *ad hoc* (IBSS) operation. Devices should assume infrastructure mode as the default.

7.2.3.2.6 NC 802.11 – Wi-Fi conformance

If 802.11 is supported, the implementation must conform to Wi-Fi test plan requirements at the time the product is offered to the market.

M	R	n/a	n/a	n/a	[7] [8] [9]

Wi-Fi is the industry consortium that does 802.11 compatibility testing. Wi-Fi interoperability requirements are increasing with time as new capabilities and features are specified by IEEE 802.11. When these capabilities are added to the Wi-Fi certification test plans, wireless implementations must conform to them.

7.2.3.3 General capability requirements – Bluetooth

7.2.3.3.1 NC Bluetooth – Base

If Bluetooth is supported, devices must implement either Bluetooth 1.1 or 1.2. In addition, devices must implement PAN profile 1.0.

M	R	n/a	n/a	n/a	[3] [4]

The PAN profile provides IP connectivity over Bluetooth.

If Bluetooth is supported, devices should implement Bluetooth 1.2.

S	R	n/a	n/a	n/a	[3] [5]

Bluetooth version 1.2 provides features such as auto-detecting 802.11; therefore, this would limit interference for 802.11.

7.2.3.3.2 NC Bluetooth – Baseband multi-slot operation

If Bluetooth is supported, Bluetooth multi-slot packet types must be supported. Supported Bluetooth packet types are DM1, DH1, DM3, DH3, DM5 and DH5.

M	R	n/a	n/a	n/a	[4] [5]

Data rate can be increased by aggregating timeslots during transmission. Utilization of such mechanisms to improve user experience is not specified by the guidelines and is left to the vendors.

7.2.3.3.3 NC Bluetooth – LMP support

7.2.3.3.3.1 If Bluetooth is supported, LMP power control message types must be supported.

M	R	n/a	n/a	n/a	[4] [5]

LMP power control message types are:

- LMP_incr_power_req;
- LMP_decr_power_req;
- LMP_max_power;
- LMP_min_power.

7.2.3.3.3.2 If Bluetooth is supported, the full range of “sniff” LMP parameters must be supported.

M	R	n/a	n/a	n/a	[4]

This will ensure that the Bluetooth link will be able to enter the “sniff” mode with the parameters requested by a MHD, as specified in these guidelines.

7.2.3.3.4 NC Bluetooth – Connection establishment

7.2.3.3.4.1 The PAN profile specifies the process of Bluetooth connection establishment between a PANU (MHD) and a NAP (M-NCF). Guidelines on M-NCF and MHD device requirements specify the M-NCF to be a NAP and the MHD device as a PANU.

7.2.3.3.4.2 If Bluetooth is supported, the NAP and PANU must cooperate to establish a Bluetooth connection as specified by the PAN profile.

M	R	n/a	n/a	n/a	[2] [4]

7.2.3.3.4.3 If Bluetooth is supported, the PANU device should initiate the Bluetooth connection to the NAP.

S	L	n/a	n/a	n/a	[2] [4]

7.2.3.3.4.4 If Bluetooth is supported, the NAP may initiate a connection to a device that advertises a PANU service record.

O	L	n/a	n/a	n/a	[2] [4]

7.2.3.3.5 NC Bluetooth – Bluetooth security mode

7.2.3.3.5.1 Bluetooth security modes are as follows:

- a) non-secure;
- b) service-level enforced security;
- c) link-level enforced security.

PAN profile service level enforced security mode consists of PAN profile authorization mode and PAN profile secrecy mode.

Recommendation of use could be reflected in the selection of device default values.

7.2.3.3.5.2 If Bluetooth is supported, Bluetooth Security Mode 2 (service-level enforced security) must be supported.

M	R	n/a	n/a	n/a	[2] [4]

7.2.3.3.5.3 If Bluetooth is supported, Bluetooth Security Mode 2 (service-level enforced security) should be used.

S	L	n/a	n/a	n/a	[2] [4]

7.2.3.3.6 NC Bluetooth – Bluetooth authentication and pairing

7.2.3.3.6.1 If Bluetooth is supported, when device authentication is necessary, the Bluetooth authentication procedure as specified in [4] Part C, 3.2, must be used.

M	R	n/a	n/a	n/a	[4]

7.2.3.3.6.2 If Bluetooth is supported, when device pairing is necessary, Bluetooth pairing procedure as specified in [4] Part C, 3.3, must be used.

M	R	n/a	n/a	n/a	[4]

7.2.3.3.6.3 If Bluetooth is supported, the "pairable" mode must be supported.

M	R	n/a	n/a	n/a	[4]

If Bluetooth is supported, the Bluetooth PIN code used for pairing must contain only numeric characters: "0-9".

M	L	n/a	n/a	n/a	[4]

Some devices do not support non-numeric PIN codes.

7.2.3.3.7 NC Bluetooth – Bluetooth PAN authentication

7.2.3.3.7.1 PAN Profile authorization modes are as follows:

- 1) open PAN;
- 2) authentication required;
- 3) authorization and authentication required.

Recommendation is that manufacturers should enable security features by default. Higher layer security mechanisms, for example, 802.1x or others may also be used as described in the BT PAN specification.

7.2.3.3.7.2 If Bluetooth is supported, then the PAN Profile v1.0 Authorization Mode 3 (authentication and authorization required) at the Bluetooth level must be supported.

M	R	n/a	n/a	n/a	[2] [4]

7.2.3.3.7.3 If Bluetooth is supported, then the PAN Profile v1.0 Authorization Mode 3 (authentication and authorization required) at the Bluetooth level should be used when connecting.

S	L	n/a	n/a	n/a	[2] [4]

7.2.3.3.8 NC Bluetooth – Bluetooth PAN encryption

7.2.3.3.8.1 PAN profile secrecy modes are as follows:

- 1) clear mode;
- 2) encrypted mode (at either the baseband or service level).

Recommendation of use could be reflected in the selection of device default values.

Supporting means that capability must be implemented and available but not necessarily used.

7.2.3.3.8.2 If Bluetooth is supported, then the PAN Profile v1.0 Secrecy Mode 2 (encrypted mode) on the Bluetooth baseband level must be supported.

M	R	n/a	n/a	n/a	[2] [4]

7.2.3.3.8.3 If Bluetooth is supported, then the PAN Profile v1.0 Secrecy Mode 2 (encrypted mode) on the Bluetooth baseband level should be used when communicating.

S	L	n/a	n/a	n/a	[4]

7.2.3.3.9 NC Bluetooth – BTH link key length requirement

If Bluetooth is supported, then the Bluetooth encryption key length must be between 8 and 16 octets.

M	L	n/a	n/a	n/a	[4]

This is the Bluetooth baseband level encryption key.

7.2.3.3.10 NC Bluetooth – DLNAQOS tolerance

If Bluetooth is supported, then incoming tagged packets must be tolerated. Tagged packets are Ethernet packets encapsulated using BNEP that include priority tags conformant with IEEE 802.3, 3.5. Here, “tolerate” means that the packet payload of any received tagged or untagged packet must be properly passed up to the higher layers in the network stack.

M	R	n/a	n/a	n/a	IEEE 802.3 [1]

The BNEP specification states that these tags must be properly carried over BNEP. It should be noted that Bluetooth devices are not required to enforce QoS over the Bluetooth link.

7.2.3.3.11 NC-PS Modes – Bluetooth specific mapping

7.2.3.3.11.1 If Bluetooth is supported, then the “active” NC-PS mode of the connection between a PANU and NAP must only be mapped to the existence of a Bluetooth link in the “active” Bluetooth mode.

M	A	n/a	n/a	n/a	[4] [5]

The NC-PS modes are bearer-independent definitions and are instantiated (mapped) differently, depending on the specific bearer used.

7.2.3.3.11.2 If Bluetooth is supported, then the “standby” NC-PS mode of the connection between a PANU and NAP must only be mapped to the existence of a Bluetooth link in the “sniff” Bluetooth low power mode.

M	A	n/a	n/a	n/a	[4] [5]

This guideline specifies how this mapping is done for Bluetooth.

7.2.3.3.11.3 If Bluetooth is supported, then the “disconnected” NC-PS mode of the connection between a PANU and NAP must only be mapped to the absence of a Bluetooth link between them.

M	A	n/a	n/a	n/a	[4] [5]

7.2.3.3.11.4 If Bluetooth is supported, then Bluetooth low power modes not mapped to any NC-PS mode, as specified in this guideline, should not be used for connectivity to the home network.

S	A	n/a	n/a	n/a	[4] [5]

The interoperability guidelines does not include other Bluetooth low power modes (i.e. “hold” and “park”) as part of the NC-PS modes scheme for Bluetooth. Their use by devices when connecting to the home network is discouraged, as it may lead to behaviour non-compliant with some of these guidelines.

7.2.3.4 General capability requirements – Network connectivity power savings (NC-PS)

7.2.3.4.1 NC-PS modes

7.2.3.4.1.1 If the NC-PS bearer level power saving scheme is supported, then there must be support for the “active” and “disconnected” NC-PS modes, as specified in these guidelines for the bearer used.

M	A	n/a	n/a	n/a	n/a

Devices must be able to map NC-PS modes to the corresponding bearer-specific modes, as specified in these guidelines.

The NC-PS modes are mapped differently depending on the specific bearer used.

7.2.3.4.1.2 If the NC-PS bearer level power saving scheme is supported, then infrastructure devices that are used to connect other devices to the home network must support the “standby” NC-PS mode, as specified in these guidelines for the bearer used.

M	A	n/a	n/a	n/a	n/a

7.2.3.4.1.3 If the NC-PS bearer level power saving scheme is supported, then devices that use infrastructure devices to connect to the home network should support the “standby” NC-PS mode, as specified in these guidelines for the bearer used.

S	A	n/a	n/a	n/a	n/a

7.2.3.4.1.4 If the NC-PS bearer level power saving scheme is supported, then devices must be able to make available the current NC-PS mode information of the connection between them.

M	A	n/a	n/a	n/a	n/a

The devices must have a common understanding of the current NC-PS mode of the connection between them.

This allows entities, such as applications or UPnP-level agents/proxies, to know the current NC-PS mode.

7.2.3.4.2 NC-PS modes – Transition control

7.2.3.4.2.1 If the NC-PS bearer level power saving scheme is supported, then transitions between the “active” and “standby” NC-PS modes must be initiated only by the device that uses an infrastructure device to connect to the home network and not by that infrastructure device.

M	A	n/a	n/a	n/a	n/a

The mobile device, as the most power-constrained end of the two, knows best when the connection should be in a power savings mode and must be in control.

7.2.3.4.2.2 If the NC-PS bearer level power saving scheme is supported, then both ends of a link must be able to force transition to the “disconnected” NC-PS mode at any time, by tearing down the link of the underlying bearer.

M	A	n/a	n/a	n/a	n/a

Nothing can prevent the underlying bearer link to be broken (on purpose or accidentally) by either end of the connection. The “disconnected” NC-PS mode is entered implicitly upon disconnection.

7.2.3.4.2.3 If the NC-PS bearer level power saving scheme is supported, then both ends of a communication link must decide that the transition to the “disconnected” NC-PS mode is completed upon detecting the loss of the link of the underlying bearer.

M	A	n/a	n/a	n/a	n/a

There is no need for any request and reply for the transition to the “disconnected” NC-PS mode.

7.2.3.4.2.4 If the NC-PS bearer level power saving scheme is supported, then devices should be able to allow control of the permissible, under these guidelines, transitions between NC-PS modes.

S	A	n/a	n/a	n/a	n/a

This allows entities, such as applications or UPnP-level agents/proxies, to control the allowed transitions between NC-PS modes.

7.2.3.4.3 NC-PS Modes – Control of bearer PS parameters

7.2.3.4.3.1 The type and range of parameters are different for each bearer. To ensure interoperability, minimum range requirements are specified in these guidelines for supported bearers.

7.2.3.4.3.2 If the NC-PS bearer level power saving scheme is supported, then any relevant parameters of the power savings mechanism of the underlying bearer must be requested by the device that uses an infrastructure device to connect to the home network, from that infrastructure device.

M	A	n/a	n/a	n/a	n/a

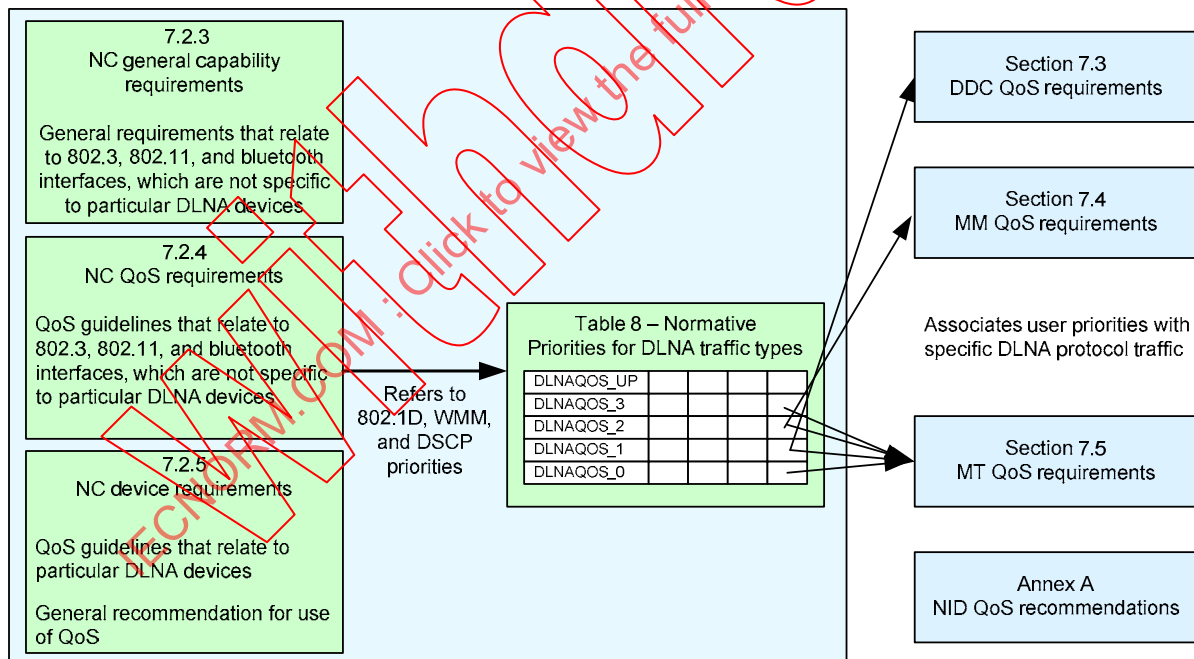
7.2.3.4.3.3 If the NC-PS bearer level power saving scheme is supported, then infrastructure devices that are used to connect other devices to the home network must respond positively to the parameters requested by these devices.

M	A	n/a	n/a	n/a	n/a

7.2.4 Networking and Connectivity – QoS requirements

7.2.4.1 General

The guidelines in 7.2.4 provide requirements for priority-based QoS, hereinafter referred to as DLNAQOS. With DLNAQOS, applications label (tag) packets with the user priority (UP) that dictates how the packets are allowed to access the network media and device queues. The DLNAQOS guidelines are contained in several subclauses as illustrated in Figure 15. Table 8 summarizes the default DLNAQOS (user priority) tag correlation between specific DLNA traffic types and different network media types. In this table, streaming, interactive, and background transfers are as defined in Table 22. The default priorities, or lower, are used if DLNAQOS is implemented. It is not permitted to use priorities above the default stated.



IEC 1583/07

Figure 15 – DLNA QoS visual organization

While there are eight priority categories defined by IEEE 802.1D, Annex G, Wi-Fi WMM [10], [11] defines only four categories. Table 8 directly correlates IEEE 802.1Q, WMM, and DSCP [38] tags without any overlap, i.e. multiple 802.1Q/DSCP values for a single WMM access class.

Table 8 – Normative priorities for DLNA traffic types

DLNAQOS_UP	DLNA traffic types	802.1Q user priority	WMM access category	DSCP	Clause details
DLNAQOS_3 (highest)	RTCP messages generated by content receivers	7	VO	0x38	7.5.6.2.18
DLNAQOS_2	Audio only or A/V streaming transfers UPnP AVtransport stream control RTCP messages generated by content sources RTSP messages	5	VI	0x28	7.5.4.2.12 7.4.1.6.22 7.5.6.2.18 7.5.8.2.80
DLNAQOS_1	Default priority for any traffic defined by DLNA guidelines, unless specified otherwise Interactive transfers	0	BE	0x00	7.3.3.7 7.5.4.2.11
DLNAQOS_0 (lowest)	Background transfers	1	BK	0x08	7.5.4.2.10

7.2.4.2 DLNAQOS Requirements – Ethernet

7.2.4.2.1 NC Ethernet DLNAQOS – Conformance

If DLNAQOS is supported on an Ethernet network interface, then it must be compliant with all mandatory requirements for tagging where tagged packets are Ethernet packets that include priority tags conformant with IEEE 802.3, 3.5, and section 9 of IEEE 802.1Q.

M	R	n/a	n/a	n/a	IEEE 802.1Q IEEE 802.3

Packet tagging is the QoS mechanism used for wired networks.

7.2.4.2.2 NC Ethernet DLNAQOS – Tagging

7.2.4.2.2.1 If DLNAQOS is supported on an Ethernet network interface, then the implementation must apply both the 802.1Q VLAN priority tag (except as noted in 0) as well as the DSCP tag to outgoing traffic in accordance with the DLNAQOS_UP value in Table 8 or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3).

M	R	n/a	n/a	n/a	IEEE 802.1Q IEEE 802.3

It is not permitted to use priorities above the values specified in Table 8.

For exchanges involving a request and response, the implementation returning a response may not know the required DLNAQOS_UP value until it has parsed the request. It is at that time when it should apply the appropriate DLNAQOS_UP value. It should be noted that there may be TCP network traffic during connection establishment that uses an inappropriate DLNAQOS_UP value.

For an HTTP streaming operation, the server must ensure the appropriate DLNAQOS_UP value (or lower) is used for the entity body.

7.2.4.2.2.2 The phrase "or a lower DLNAQOS_UP" value means that the highest permitted DLNAQOS_UP value should be used.

S	A	n/a	n/a	n/a	n/a

7.2.4.2.2.3 The phrase "or a lower DLNAQOS_UP" value also means that a lower DLNAQOS_UP value (indicated in the guideline) may be used.

O	A	n/a	n/a	n/a	n/a

7.2.4.2.2.4 For best-effort traffic on Ethernet, the implementation may simply omit the 802.1Q VLAN tag because frames with no tag are handled best-effort by default.

O	R	n/a	n/a	n/a	IEEE 802.1Q IEEE 802.3

7.2.4.3 DLNAQOS Requirements – 802.11

7.2.4.3.1 NC 802.11 DLNAQOS – Conformance

If DLNAQOS is supported on an 802.11 network interface, then it must conform to all mandatory requirements in the WiFi WMM Test Plan [11] and specification [10].

M	R	n/a	n/a	n/a	[10] [11]

QoS support is optional, but if supported, it must conform to Wi-Fi requirements.

WMM provides the base level QoS specification for 802.11 network devices.

7.2.4.3.2 NC 802.11 DLNAQOS – Tagging

If DLNAQOS is supported on an 802.11 network interface, then the implementation must apply both the WMM tag as well as the DSCP tag to outgoing traffic in accordance with the appropriate DLNAQOS_UP value in Table 8 or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3).

M	R	n/a	n/a	n/a	IEEE 802.1Q [10] [11] [38]

802.1Q and DSCP correlation is in agreement with the WMM test plan recommendation.

It is not permitted to use priorities above the values specified in Table 8.

For exchanges involving a request and response, the implementation returning a response may not know the required DLNAQOS_UP value until it has parsed the request. It is at that time when it should apply the appropriate DLNAQOS_UP value. It should be noted that there may be TCP network traffic during connection establishment that uses an inappropriate DLNAQOS_UP value.

7.2.4.4 DLNAQOS Requirements – Bluetooth

7.2.4.4.1 NC Bluetooth DLNAQOS: Conformance

If DLNAQOS is supported on a Bluetooth network interface, then it must be compliant with all mandatory requirements for tagging Ethernet traffic which is encapsulated by BNEP, where tagged packets are Ethernet packets that include priority tags conformant with IEEE 802.3, 3.5, and section 9 of IEEE 802.1Q.

M	R	n/a	n/a	n/a	IEEE 802.1Q IEEE 802.3 [1]

Packet tagging is the QoS mechanism used for wired networks. BNEP specifies how tagged Ethernet packets are encapsulated.

7.2.4.4.2 NC Bluetooth DLNAQOS – Tagging

7.2.4.4.2.1 If DLNAQOS is supported on a Bluetooth network interface, then the implementation must apply both to the 802.1Q VLAN priority tag (except as noted in 7.2.4.4.2.2) as well as to the DSCP tag to outgoing Ethernet traffic which is encapsulated by BNEP, in accordance with the DLNAQOS_UP value in Table 8 or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3).

M	R	n/a	n/a	n/a	IEEE 802.1Q IEEE 802.3 [1]

It is not permitted to use priorities above the values specified in Table 8.

For exchanges involving a request and response, the implementation returning a response may not know the required DLNAQOS_UP value until it has parsed the request. It is at that time when it should apply the appropriate DLNAQOS_UP value.

NOTE There may be TCP network traffic during connection establishment that uses an inappropriate DLNAQOS_UP value.

For an HTTP streaming operation, the server must ensure the appropriate DLNAQOS_UP value (or lower) is used for the entity body.

7.2.4.4.2.2 If DLNAQOS is supported on a Bluetooth network interface, for best-effort Ethernet traffic, which is encapsulated by BNEP, then the implementation may simply omit the 802.1Q VLAN tag because frames with no tag are handled best-effort by default.

O	R	n/a	n/a	n/a	IEEE 802.1Q IEEE 802.3 [1]

Many legacy network devices do not handle 802.1Q tags well (do not expect a larger frame header) and react unfavourably. Not marking best-effort traffic allows for the greatest interoperability with these devices.

The DSCP value for the best effort traffic is 0 anyway.

802.1Q and DSCP correlation is in agreement with the WMM test plan recommendation.

7.2.5 Networking and connectivity – Device requirements

7.2.5.1 General

Subclause 7.2.5 specifies the capabilities or combination of capabilities that DLNA devices specifically support. These specific device requirements reference the guidelines defined in 7.2.3, Table 8, and 7.2.4. For example, a specific requirement for DLNA HND devices is that they must support either Ethernet or 802.11 connectivity, where Ethernet and 802.11 capabilities are described in 7.2.3 and are referenced by name. Correspondingly, a specific requirement for DLNA MHD devices is that they must support either Ethernet, 802.11, or Bluetooth connectivity, where Ethernet, 802.11 and Bluetooth capabilities are described in 7.2.3 and are referenced by name.

7.2.5.2 Device requirements – Common

7.2.5.2.1 NC devices – IP stack

DLNA device classes must support a TCP/IP stack that includes IPv4, TCP, UDP, ARP, and ICMP components conformant to all required client aspects of [22].

M	R	HND	MHD	MIU	[17] [18] [19] [20] [21] [22]

A DNS client is omitted because it is not strictly needed for UPnP operations on the network. Native IP addresses actually simplify the use of UPnP.

7.2.5.2.2 NC devices – IP address acquisition

7.2.5.2.2.1 DLNA device classes must support DHCP client functionality [28] and obtain an IP address and subnet mask from a home network DHCP server if present. They must implement Auto-IP as defined by the UPnP device architecture v1.0 specification ([13], [53]) so that if a DHCP server is not present on the home network, a link-local network address may be automatically acquired.

M	R	HND	MHD	MIU	[13] [28] [53]

This guideline includes an assumption that devices must transition from DHCP to Auto-IP, when devices fail to renew an expired IP address (that was assigned by a DHCP server). Likewise, Auto-IP requires the devices to make periodic attempts (once every 5 min) to acquire a DHCP-assigned IP address, when operating with a self-assigned IP address.

Although it is desirable for a device using a DHCP-assigned IP address to exhibit interoperability with a device using an Auto-IP address, DLNA makes no requirement that full interoperability will occur in such scenarios.

7.2.5.2.2.2 DLNA device classes that allocate addresses with the Auto-IP link local address allocation system of [13] should adhere to all aspects of 2.4 of [13]. No packet generated by a DLNA device class should be sent to a router for forwarding.

S	R	HND	MHD	MIU	[13]

Section 2 of [13] contains information on how Auto-IP addresses are impacted by routed subnets.

7.2.5.2.2.3 DLNA device classes that allocate addresses with the DHCP address allocation system should adhere to all aspects of 2.4 of [13]. No packet generated by a DLNA device class that has an Auto-IP allocated source address should be sent to a router for forwarding.

S	R	HND	MHD	MIU	[13]

Section 2 of [13] contains information on how Auto-IP addresses are impacted by routed subnets.

7.2.5.2.2.4 DLNA device classes that allocate addresses with the Auto-IP link local address allocation system of [13] should attempt to interoperate with DLNA device classes that have allocated DHCP addresses according to section 4 of [13] as to the interaction between link-local and non-link-local addresses. A DLNA device class should attempt to send all packets on the local link.

S	R	HND	MHD	MIU	[13]

Section 4 of [13] contains information about multiple addresses per interface; however, it also contains a paragraph on the interoperability of link-local and non-link local addresses. The relevant section follows.

Thus a host with only a link-local address wishing to send a packet to another host on the same link with only a non-link-local address MAY use ARP on the link to find the hardware address and send the packet directly to the destination host. (This is equivalent to saying that a host with only a link-local address should have a default route entry indicating that all hosts are directly reachable through its IP interface, without going through any gateway.) Likewise, a host with only a non-link-local address wishing to send a packet to a link-local destination address MAY use ARP and send the packet directly to the destination host on that link. (This is equivalent to saying that a host with only a non-link-local address should also have a route entry indicating that net 169.254/16 is directly reachable through its IP interface, without going through any gateway.) This allows a host which has only a link-local address to communicate with another host on the same link which has only a non-link-local address.

7.2.5.2.2.5 DLNA device classes that allocate addresses with the DHCP address allocation system should attempt to interact with endpoints that have allocated link-local Auto-IP addresses according to section 4 of [13]. A DLNA device class should attempt to send all packets bound to a link-local address on the local link.

S	R	HND	MHD	MIU	[13]

7.2.5.2.2.6 When the lease on an IP address expires, and the DLNA device class is unable to renew the lease on that IP address, or obtain a lease on a new IP address, the device class must obtain an Auto-IP address as defined by the UPnP device architecture v1.0.

M	A	HND	MHD	MIU	[13] [28] [53]

DLNA device classes must not utilize an expired DHCP IP address.

7.2.5.2.3 NC devices – DLNAQOS support

7.2.5.2.3.1 DLNA device classes should support DLNAQOS on all network interfaces.

S	A	HND	MHD	MIU	n/a

7.2.5.2.3.2 If DLNAQOS is supported on an Ethernet network interface by a DLNA device class, then it must be conformant to all [NC Ethernet DLNAQOS:] labelled requirements in 7.2.4.

M	A	HND	MHD	MIU	n/a

7.2.5.2.3.3 If DLNAQOS is supported on an 802.11 network interface by a DLNA Device Class, then it must be conformant to all [NC 802.11 DLNAQOS:] labelled requirements in 7.2.4.

M	A	HND	MHD	MIU	n/a

7.2.5.2.3.4 If DLNAQOS is supported on a Bluetooth network interface by a DLNA device class, then it must be conformant to all [NC Bluetooth DLNAQOS:] labelled requirements in 7.2.4.

M	A	MHD	n/a	MIU	n/a

7.2.5.3 Device requirements – HND

7.2.5.3.1 NC HND devices – Required connectivity

DLNA device classes must support at least one of the following connectivity selections.

- Ethernet conformant to all [NC Ethernet:] labelled requirements in 7.2.3.
- 802.11 conformant to all [NC 802:11:] labelled requirements in 7.2.3.

M	R	HND	n/a	MIU	n/a

7.2.5.3.2 NC HND devices – Recommended connectivity

DLNA device classes should support all of the following connectivity selections.

- Ethernet conformant to all [NC Ethernet:] labelled requirements in 7.2.3.
- 802.11 conformant to all [NC 802:11:] labelled requirements in 7.2.3.

Any of the above selections can be supported via an add on card, dongle, or equivalent.

S	R	HND	n/a	MIU	n/a

This guideline is intended to ensure that a consumer does not have to understand the different network connectivity types when purchasing a DLNA product. A consumer will be assured a newly purchased product will work with other previously purchased DLNA products.

7.2.5.4 Device requirements – MHD

7.2.5.4.1 NC MHD devices – Required connectivity

DLNA device classes must support at least one of the following connectivity selections.

- Ethernet conformant to all [NC Ethernet:] labelled requirements in 7.2.3.
- 802.11 conformant to all [NC 802.11:] labelled requirements in 7.2.3.
- Bluetooth conformant to all [NC Bluetooth:] labelled requirements in 7.2.3.

M	R	MHD	n/a	MIU	n/a

7.2.5.4.2 NC MHD devices – Connection to the home network

If an M-NCF device class is detected for the desired bearer, a DLNA device class should use it to connect to the home network.

S	A	MHD	n/a	MIU	n/a

It is up to the MHD (for example, based on user or application preferences) to select the desired bearer to connect to the home network.

Connectivity for the desired bearer via an M-NCF, when possible, provides a better user experience and is recommended. This recommendation could be reflected, for example, in the MHD's default settings.

Direct connectivity for the desired bearer can also occur via standard networking elements such as access points, switches or hubs, etc.

7.2.5.4.3 NC MHD devices – Topology restriction

A DLNA device class must not be connected to more than one M-NCF at a time.

M	A	MHD	n/a	MIU	n/a

To avoid using the spanning tree protocol in M-NCF.

7.2.5.4.4 NC MHD devices – Bluetooth device discovery

7.2.5.4.4.1 If Bluetooth is supported, then a DLNA device class must be able to perform Bluetooth inquiry.

M	R	MHD	n/a	MIU	[4]

MHD enters the inquiry substate to perform Bluetooth device discovery.

7.2.5.4.4.2 If Bluetooth is supported, then a DLNA device class should filter inquiry responses based on the networking bit in the class of device field to narrow down the search for M-NCFs.

S	L	MHD	n/a	MIU	[3] [4]

It is recommended for MHDs to reduce Bluetooth device discovery time.

7.2.5.4.5 NC MHD devices – Bluetooth SDP

7.2.5.4.5.1 If Bluetooth is supported, then a DLNA device class must be able to retrieve the SDP record from an NAP.

M	R	MHD	n/a	MIU	[3] [4]

M-NCF is a NAP.

7.2.5.4.5.2 If Bluetooth is supported, then a DLNA device class should not expose the 'M-NCF_Description' text portion of the ServiceDescription attribute specified in 7.2.5.5.15.5, in order to provide a better user experience.

S	A	MHD	n/a	MIU	[3] [4]

M-NCF advertises its specific DLNA capabilities that can be shown as BT device description. This specific description part is not in user friendly form and should therefore not be shown.

7.2.5.4.6 NC MHD devices – MHD Bluetooth PIN

If Bluetooth is supported, then a DLNA device class must implement support for variable PIN codes.

M	L	MHD	n/a	MIU	[4]

The use of fixed PIN in both the MHD device and M-NCF would preclude interoperability.

7.2.5.4.7 NC MHD devices – BT PAN role

If Bluetooth is supported, then an MHD device class must perform the role of a PANU.

M	L	MHD	n/a	MIU	[3]

7.2.5.4.8 NC MHD devices – MHD device PAN compressed packet type

If Bluetooth is supported, then BNEP Ethernet header compression (BNEP_COMPRESSED_ETHERNET_DEST_ONLY Packet type) should be used.

S	L	MHD	n/a	MIU	[3]

Reduces amount of redundant header information sent over point-to-point link.

7.2.5.4.9 NC MHD devices – BT PS mode support

If Bluetooth is supported, then the “sniff” Bluetooth power savings mode should be supported.

S	R	MHD	n/a	MIU	[4]

7.2.5.4.10 MHD devices – NC-PS mode support using Bluetooth

7.2.5.4.10.1 If Bluetooth is supported, then a DLNA device class must support the NC-PS bearer level power saving scheme conformant to all [NC-PS modes:] labelled requirements in 7.2.3.

M	A	MHD	n/a	MIU	n/a

MHD devices use infrastructure devices to connect to the home network and as such they should support the "standby" NC-PS mode, as specified in 7.2.3.4.1.3.

7.2.5.4.10.2 If Bluetooth is supported, then a DLNA device class should support the "standby" NC-PS mode.

S	A	MHD	n/a	MIU	n/a

7.2.5.4.10.3 If Bluetooth is supported, then a DLNA device class must be the one to initiate the transition from "active" to "standby" NC-PS mode, by requesting to move the Bluetooth link from the "active" to the "sniff" Bluetooth mode, following the "sniff" parameter restrictions specified in 7.2.5.4.11.

M	A	MHD	n/a	MIU	[4]

7.2.5.4.10.4 If Bluetooth is supported, then a DLNA device class must be the one to request at any time the transition from "standby" to "active" NC-PS mode, by sending an LMP request to move the link from "sniff" to "active" Bluetooth mode.

M	A	MHD	n/a	MIU	[4]

7.2.5.4.10.5 If Bluetooth is supported, while the connection is in "standby" NC-PS mode, then a change in parameters of the "sniff" Bluetooth power savings mode can be requested at any time, following the "sniff" parameter restrictions specified in 7.2.5.4.11.

O	A	MHD	n/a	MIU	[4]

Requesting a change of the parameters of the current "sniff" Bluetooth power savings mode does not trigger a transition out of the "standby" mode.

7.2.5.4.11 NC MHD Devices – Requested "sniff" parameter restrictions

7.2.5.4.11.1 If Bluetooth is supported, then all "sniff" parameter requests sent from a DLNA device class to the M-NCF must meet the restrictions outlined in this guideline.

M	A	MHD	n/a	MIU	[4]

7.2.5.4.11.2 If Bluetooth is supported and a DLNA device class determines from the M-NCF_Description text specified in 7.2.5.5.15.5 that the M-NCF supports ARP proxying, then any requested HCI Sniff_Max_Interval value must be less than, or equal to, 20 s.

M	A	MHD	n/a	MIU	[4]

7.2.5.4.11.3 If Bluetooth is supported and if a DLNA device class determines from the M-NCF_Description text specified in 7.2.5.5.15.5, the M-NCF does not support "ARP proxying", then any requested HCI Sniff_Max_Interval value must be less than, or equal to, 2 s.

M	A	MHD	n/a	MIU	[3] [4]

7.2.5.4.11.4 If Bluetooth is supported, any HCI Sniff_Min_Interval request must be less than, or equal to, the HCI Sniff_Max_Interval.

M	R	MHD	n/a	MIU	[4]

7.2.5.4.11.5 If Bluetooth is supported and if a DLNA device class determines from the M-NCF_Description text specified in 7.2.5.5.15.5 that the M-NCF supports "ARP proxying", then any HCI Sniff_Attempt request should be greater than, or equal to, 625 ms.

S	A	MHD	n/a	MIU	[3] [4]

This will ensure a minimum duty cycle while the BTH link between MHD and NCF is in the "sniff" mode and, therefore, a minimum maintained bandwidth.

7.2.5.4.11.6 If Bluetooth is supported and if a DLNA device class determines from the M-NCF_Description text specified in 7.2.5.5.15.5 that the M-NCF does not support "ARP proxying", then any HCI Sniff_Attempt request should be greater than, or equal to, 62,5 ms.

S	A	MHD	n/a	MIU	[4]

7.2.5.4.11.7 If Bluetooth is supported, any HCI Sniff_Timeout request must be greater than, or equal to, the HCI Sniff_Max_Interval.

M	R	MHD	n/a	MIU	[4]

7.2.5.4.12 NC MHD devices – Bluetooth establishing network access rights

7.2.5.4.12.1 If Bluetooth is supported a DLNA device class must be able to initiate a request for network access rights from an M-NCF by initiating a pairing procedure with the M-NCF, as specified in 7.2.3.3.6

M	R	MHD	n/a	MIU	[4]

Network access right is the pre-approval for future network connection requests from an MHD. Connection requests from an MHD with permanent network access rights are always granted. Connection requests from an MHD with temporary network access rights are granted if other additional conditions are satisfied.

Network access rights must be established individually because each M-NCF has its own Bluetooth device database.

Pairing may be initiated either by the MHD or the M-NCF.

7.2.5.4.12.2 If Bluetooth is supported, any DLNA device class may initiate the Bluetooth pairing process to establish network access rights.

O	R	MHD	n/a	MIU	[4]

7.2.5.4.12.3 If Bluetooth is supported and multiple M-NCFs are in range, then a DLNA device class must establish network access rights with each M-NCF individually.

M	A	MHD	n/a	MIU	[4]

7.2.5.4.13 NC MHD Devices – Link key usage and management

If Bluetooth is supported and if there is a record of the DLNA device class in the M-NCF's Bluetooth device database, then the DLNA device class and the M-NCF must use the link key stored in the database for authentication and deriving the encryption key.

M	A	MHD	n/a	MIU	[1] [4]

7.2.5.5 Device requirements – M-NCF

7.2.5.5.1 NC M-NCF – Required connectivity

An M-NCF must have network interfaces to connect to the following two connectivity domains:

- the home connectivity domain with the bearers specified in 7.2.5.3.1;
- the mobile connectivity domain with Bluetooth conformant to all [NC Bluetooth:] labelled requirements in 7.2.3.

M	A	n/a	n/a	M-NCF	n/a

The M-NCF bridges the connectivity gap between the HND and MHD connectivity domains.

The current version of guidelines specifies only a BT M-NCF.

7.2.5.5.2 NC M-NCF – Bridging

7.2.5.5.2.1 An M-NCF must perform layer-two bridging between home-connectivity domain-bearer interfaces and the mobile-connectivity domain-bearer interfaces according to 802.1D, as further specified by the PAN profile [3].

M	A	n/a	n/a	M-NCF	[3] [6]

Filtering of packets at the M-NCF may be allowed for power saving.

7.2.5.5.2.2 An M-NCF must not do any additional filtering to that specified in [6] and 7.2.5.5.9.

M	A	n/a	n/a	M-NCF	[3] [6]

7.2.5.5.3 NC M-NCF – IP protocol stack

An M-NCF should support a TCP/IP stack that includes IPv4, TCP, UDP, ARP, and ICMP components conformant to all required client aspects of [22].

S	R	n/a	n/a	M-NCF	[17] [18] [19] [20] [21] [22]

TCP/IP stack support is not mandatory for M-NCF operations, but it can be used for proving device management, for example, through SNMP or web-based interface

A DNS client is omitted because it is not strictly needed for UPnP operations on the network. Native IP addresses actually simplify the use of UPnP.

7.2.5.5.4 NC M-NCF – IP address acquisition

An M-NCF should support item 7.2.5.2.2

S	R	n/a	n/a	M-NCF	[13] [28] [53]

7.2.5.5.5 NC M-NCF – Multiple device support

An M-NCF must allow connections to multiple active client devices simultaneously.

M	A	n/a	n/a	M-NCF	n/a

7.2.5.5.6 NC M-NCF – ARP proxying functionality

7.2.5.5.6.1 An M-NCF should support ARP proxying functionality as specified in the PAN profile [3] and further clarified in this guideline.

S	C	n/a	n/a	M-NCF	[3]

7.2.5.5.6.2 If an M-NCF supports the ARP proxying functionality, then it must do so as follows.

- It must first obtain the IP and MAC address of an MHD after the MHD is connected to it.
- After obtaining the MHD's IP and MAC address, it must stop forwarding any ARP requests to the MHD and it must reply to any ARP requests targeted at the MHD's IP address using the MHD's MAC address.

M	C	n/a	n/a	M-NCF	[3]

7.2.5.5.7 NC M-NCF – NC-PS conformance

7.2.5.5.7.1 An M-NCF must support the NC-PS bearer level power-saving scheme conformant to all [NC-PS modes:] labelled requirements in 7.2.3.

M	A	n/a	n/a	M-NCF	n/a

The M-NCF is an infrastructure device that is used to connect other devices to the home network and, as such, it must support the NC-PS mode as specified in 7.2.3.4.1.2.

7.2.5.5.7.2 An M-NCF must support the “standby” NC-PS mode.

M	A	n/a	n/a	M-NCF	n/a

7.2.5.5.8 NC M-NCF – NC-PS connection individuality

An M-NCF must allow each connection to an MHD to be in a separate NC-PS mode, independently of the NC-PS mode of other connections.

M	A	n/a	n/a	M-NCF	n/a

7.2.5.5.9 NC M-NCF – NC-PS traffic reduction operations

7.2.5.5.9.1 While the connection between the DLNA device class and the MHD is in the “active” NC-PS mode, an M-NCF must not perform any traffic reduction operations that may reduce functionality at the IP-layer and above as experienced by the MHD device.

M	A	n/a	n/a	M-NCF	n/a

Examples of operations that would reduce the functionality of the MHD at the IP-layer and above are filtering of any DHCP-related traffic and filtering of any UPnP-related traffic.

7.2.5.5.9.2 While the connection between the DLNA device class and the MHD is in the “standby” mode, an M-NCF should implement traffic reduction operations to prevent any or all of the following traffic from being received by the MHD.

- UPnP multicast traffic directed to address 239.255.255.250:1900.

- ARP messages, provided that the DLNA device class also performs the “ARP-proxying” functionality specified in 7.2.5.5.6 on behalf of the MHD device.

S	A	n/a	n/a	M-NCF	[21] [53]

Reducing the traffic flowing in the link between the MHD and M-NCF results in power savings for the MHD device since the two ends need to spend less time communicating.

In “standby” mode an MHD device can receive (also send) unicast IP traffic, but may not receive UPnP multicast traffic if the M-NCF filters it. As a result, the MHD's full network state is maintained but its UPnP state may be lost.

7.2.5.5.10 NC M-NCF – Bluetooth discovery

7.2.5.5.10.1 If Bluetooth is supported, then an M-NCF should be in Bluetooth discoverable mode.

S	L	n/a	n/a	M-NCF	[4]

The M-NCF should perform a Bluetooth inquiry scan to allow for Bluetooth discovery.

7.2.5.5.10.2 If Bluetooth is supported, then an M-NCF must set the networking bit of the class of device (CoD) field.

M	R	n/a	n/a	M-NCF	[4]

Setting the CoD accelerates M-NCF Bluetooth discovery by inquiring MHDs.

7.2.5.5.11 NC M-NCF – Bluetooth connectability

If Bluetooth is supported, then an M-NCF must be in Bluetooth connectable mode.

M	L	n/a	n/a	M-NCF	[4]

The M-NCF should perform Bluetooth page scan to allow for incoming connections.

7.2.5.5.12 NC M-NCF – Bluetooth compressed packet type

If Bluetooth is supported, an M-NCF should use the BNEP Ethernet header compression (BNEP_COMPRESSED_ETHERNET_SOURCE_ONLY Packet type).

S	L	n/a	n/a	M-NCF	[3]

This reduces the amount of redundant header information sent over the point-to-point link.

7.2.5.5.13 NC M-NCF – Bluetooth PAN role

If Bluetooth is supported, then an M-NCF must perform the role of a NAP, as specified in the PAN profile.

M	L	n/a	n/a	M-NCF	[3]

7.2.5.5.14 NC M-NCF – Bluetooth PAN mode

If Bluetooth is supported, then an M-NCF must operate in the NAP multi-user mode as specified in the PAN profile.

M	L	n/a	n/a	M-NCF	[3]

This implies that a master-slave role switch upon connection may be necessary, so that the M-NCF is always the master in all Bluetooth connections.

7.2.5.5.15 NC M-NCF – Bluetooth SDP

7.2.5.5.15.1 If Bluetooth is supported, then an M-NCF must act as Bluetooth SDP server.

M	L	n/a	n/a	M-NCF	[3] [4]

7.2.5.5.15.2 If Bluetooth is supported, then an M-NCF must create a NAP service record in the service discovery database as specified in [3].

M	R	n/a	n/a	M-NCF	[3]

7.2.5.5.15.3 If Bluetooth is supported, then the NAP service record of an M-NCF must use one of the following values for the NetAccessType attribute.

- 0x0004 if the home connectivity domain bearer is 10 Mb Ethernet.
- 0x0005 if the home connectivity domain bearer is 100 Mb Ethernet.
- 0xFFFE if the home connectivity domain bearer is Ethernet at other rates, or if the HNV1 bearer is 802.11.

M	L	n/a	n/a	M-NCF	[3]

The PAN profile has not assigned attribute values to represent 802.11 or Ethernet at rates other than 10 Mb and 100 Mb as the NetAccessType.

7.2.5.5.15.4 If Bluetooth is supported, then the NAP SDP record (see [4], Part E) of an M-NCF must include in the ServiceDescription attribute value the 'M-NCF_Description' text of the form "DLNAv1.5_M-NCF_XXX_YYY", where the text options XXX and YYY are specified below, to indicate that it is a DLNA v1.5 M-NCF and its supported functionality.

- XXX is equal to "FOF", if the DLNA device class does not support the UPnP multicast filtering functionality specified in 7.2.5.5.9.2.
- XXX is equal to "FON", if the DLNA device class supports the UPnP multicast filtering functionality specified in 7.2.5.5.9.2 and will always apply filtering when the connection between an MHD and the M-NCF is in the "standby" NC-PS mode.
- YYY is equal to "POF", if the DLNA device class does not support the "ARP-proxying" functionality specified in 7.2.5.5.6;
- YYY is equal to "PON", if the DLNA device class supports the "ARP-proxy" functionality specified in 7.2.5.5.6 and will always perform the "ARP-proxying" functionality when the connection between the MHD and the DLNA Device Class is in the "standby" NC-PS mode.

M	A	n/a	n/a	M-NCF	[4]

The ServiceDescription attribute is a human-readable text containing a brief description of the Bluetooth service. The MHD will be able to tell whether a Bluetooth NAP is a DLNA compliant Bluetooth M-NCF by looking at the ServiceDescription attribute value of the NAP SDP record. The MHD will also know whether the M-NCF is filtering UPnP multicast messages or whether it is performing ARP proxying on its behalf.

7.2.5.5.15.5 If Bluetooth is supported, then an M-NCF must prevent the "M-NCF_Description" text portion of the ServiceDescription attribute value from being edited or deleted.

M	A	n/a	n/a	M-NCF	[4]

Only the M-NCF must have control of editing the "M-NCF_Description" text portion of the ServiceDescription field. It must not allow other entities, for example, users or other applications from overwriting this portion of the field.

7.2.5.5.16 NC M-NCF – Bluetooth device database definition

7.2.5.5.16.1 If Bluetooth is supported, then an M-NCF device must maintain, or have full access to, a Bluetooth device database to store security-related information on devices. The Bluetooth device database must be stored in non-volatile memory.

M	A	n/a	n/a	M-NCF	[2]

Bluetooth device database is a non-volatile storage and need not be co-located with the M-NCF. The term database is used as defined in [2] and does not refer to specific implementations, but only to a general information storage facility.

7.2.5.5.16.2 If Bluetooth is supported, then the Bluetooth device database of an M-NCF device must, at a minimum, store the following information about an MHD device:

- BD_ADDR;
- device name;
- link key;
- authorized for PAN service;
- trusted or untrusted.

M	A	n/a	n/a	M-NCF	[2] [3] [4]

"Authorized for PAN service" indicates whether future PAN connection requests from an untrusted device are automatically authorized.

NOTE The trust-level field in the device database differentiates between trusted and untrusted devices, which have fixed relationship (paired and have link key).

Unknown devices, i.e. devices without an entry at this device database must be treated as untrusted.

Database is a descriptive term, which does not say how to implement it.

7.2.5.5.16.3 If Bluetooth is supported, then the Bluetooth device database of an M-NCF should store the following information about an MHD device:

- validity time of the entry from the time of its creation.

S	A	n/a	n/a	M-NCF	n/a

The validity time specifies the duration for which temporary network access rights are granted.

7.2.5.5.16.4 If validity time is stored in the Bluetooth device database of a DLNA device class, then the validity time must be specified in seconds.

M	A	n/a	n/a	M-NCF	[2]

7.2.5.5.17 NC M-NCF – Bluetooth service database definition

7.2.5.5.17.1 If Bluetooth is supported, then an M-NCF must maintain, or have full access to, a Bluetooth service database to store security-related information on services it supports.

M	A	n/a	n/a	M-NCF	[2]

7.2.5.5.17.2 If Bluetooth is supported, then the Bluetooth service database of an M-NCF must, at a minimum, store the following information about a service they support:

- authentication is required or not;
- authorization is required or not;
- encryption is required or not.

M	A	n/a	n/a	M-NCF	[2] [3]

7.2.5.5.17.3 If Bluetooth is supported, then an M-NCF must create an entry in the Bluetooth service database for the PAN service.

M	A	n/a	n/a	M-NCF	[1] [3]

7.2.5.5.18 NC M-NCF – Bluetooth establishing network access rights

7.2.5.5.18.1 If Bluetooth is supported, then an M-NCF must be able to initiate granting of network access rights to an MHD device by initiating a pairing procedure with the MHD device, as specified in 7.2.3.3.6.

M	R	n/a	n/a	M-NCF	[4]

Network access rights are the pre-approval for future network connections from an MHD. Connection requests from an MHD with permanent network access rights are always granted.

Connection requests from an MHD with temporary network access rights are granted if other additional conditions are satisfied.

Pairing may be initiated either by the MHD or the M-NCF.

7.2.5.5.18.2 If Bluetooth is supported, then an M-NCF may initiate the pairing process to establish network access rights.

O	L	n/a	n/a	M-NCF	[4]

7.2.5.5.18.3 If Bluetooth is supported, upon successful completion of the pairing procedure with an MHD, an M-NCF must decide whether to grant permanent network access rights, temporary network access rights, or no access rights at all to that MHD.

M	A	n/a	n/a	M-NCF	n/a

7.2.5.5.18.4 If Bluetooth is supported, in order for an M-NCF to grant permanent network access rights to an MHD, it must

- create an entry into the Bluetooth Device Database consisting of the BT device address, device name, and the associated link key;
- set the device as Trusted.

M	A	n/a	n/a	M-NCF	[2]

The choice is determined by implementation-specific mechanisms including, but not limited to, fixed rules and user interactions.

7.2.5.5.18.5 If Bluetooth is supported, in order for an M-NCF to grant temporary network access rights to an MHD, it must

- create an entry into the Bluetooth device database consisting of the Bluetooth device address, device name, and the associated link key;
- set the device as untrusted;
- set "Authorized for PAN" true;
- set the validity time of the MHD device record, if the validity time field is present.

M	A	n/a	n/a	M-NCF	[2]

The choice of value for "Authorized for PAN service" is decided by implementation-specific mechanisms including, but not limited to, user interactions. When set to true, the connection request is granted after authentication. When set to false, the M-NCF may further initiate other mechanisms including, but not limited to, user interactions to grant or deny the connection request.

7.2.5.5.18.6 If Bluetooth is supported, in order for an M-NCF to grant neither permanent nor temporary network access rights to an MHD, it must not create an entry into the Bluetooth device database for this MHD.

M	A	n/a	n/a	M-NCF	[2]

7.2.5.5.18.7 If Bluetooth is supported, an M-NCF should be able to grant one time network access to an MHD without granting permanent or temporary access rights to that MHD.

S	A	n/a	n/a	M-NCF	n/a

7.2.5.5.19 NC M-NCF – Bluetooth managing network access rights

7.2.5.5.19.1 If Bluetooth is supported, an M-NCF may convert temporary network access rights granted to an MHD to permanent network access rights by setting the "trusted or untrusted" field in the Bluetooth device database as "trusted".

O	A	n/a	n/a	M-NCF	[2]

7.2.5.5.19.2 If Bluetooth is supported and if a validity time exists in the Bluetooth device database, an M-NCF may extend the duration of the validity time before the validity time expires.

O	A	n/a	n/a	M-NCF	[2]

This process can be triggered by implementation-specific mechanisms including, but not limited to, user requests.

7.2.5.5.20 NC M-NCF – Bluetooth revoking network access rights

7.2.5.5.20.1 If Bluetooth is supported, an M-NCF must be able to revoke permanent or temporary network access rights of an MHD in the Bluetooth device database at any time.

M	A	n/a	n/a	M-NCF	[1]

This process can be triggered by implementation-specific mechanisms including, but not limited to, user requests.

7.2.5.5.20.2 If Bluetooth is supported and if a validity time exists in the Bluetooth device database, an M-NCF should revoke temporary network access rights from an MHD when the validity time of that MHD expires.

S	A	n/a	n/a	M-NCF	n/a

This process can be triggered by implementation-specific mechanisms including, but not limited to, user requests.

7.2.5.5.20.3 If Bluetooth is supported, in order for an M-NCF to revoke network access rights from an MHD, it must do so by deleting the entry of the MHD from the Bluetooth device database, and terminating the MHD connection, if the connection is present.

M	A	n/a	n/a	M-NCF	[2]

7.2.5.5.21 NC M-NC – Bluetooth access control

7.2.5.5.21.1 If Bluetooth is supported, upon receiving a connection request from an MHD, an M-NCF must accept or deny the connection request according to the following guidelines 7.2.5.5.21.2 to 7.2.5.5.21.5.

M	A	n/a	n/a	M-NCF	n/a

7.2.5.5.21.2 If the PAN service record in the Bluetooth service database has the value of "authorization required" set to false, and "authentication required" is set to false, an M-NCF must accept the connection request from the MHD.

M	A	n/a	n/a	M-NCF	[2] [3]

7.2.5.5.21.3 If the PAN service record in the Bluetooth service database has the value of "authorization required" set to false, and "authentication required" set to true, the connection request must be accepted by an M-NCF after authentication is completed successfully as specified in 7.2.3.3.6.

M	A	n/a	n/a	M-NCF	[2] [3]

7.2.5.5.21.4 If the PAN service record in the Bluetooth service database has the value of "authorization required" set to True, then the connection request from MHD must be accepted by the DLNA device class in the following cases.

- If there is a record of the MHD in the Bluetooth device database and the device is marked as trusted, the connection request must be accepted by the device after authentication is completed successfully as specified in 7.2.3.3.6.
- If there is a record of the MHD in the Bluetooth device database, the device is marked as untrusted and has the value of "authorized for PAN service" set to true, the connection request must be accepted by the device after authentication is completed successfully as specified in 7.2.3.3.6.

M	A	n/a	n/a	M-NCF	[2] [3]

7.2.5.5.21.5 If the PAN service record in the Bluetooth service database has the value of "authorization required" set to true, and the Bluetooth device database does not have a record of the MHD requesting the connection, then an M-NCF may use other implementation-specific mechanisms to determine whether to accept or deny the connection request after authentication is completed as specified in 7.2.3.3.6.

O	A	n/a	n/a	M-NCF	[2] [3]

7.2.5.5.22 NC M-NCF link key usage and management

7.2.5.5.22.1 If Bluetooth is supported and there is a record of the MHD in a DLNA device class's Bluetooth device database, then the MHD device and the DLNA device class must use the link key stored in the database for authentication and deriving the encryption key for this session with the MHD.

M	A	n/a	n/a	M-NCF	[2] [3]

Implementation-specific mechanisms may include, but are not limited to, fixed rules, user interaction or other trusted external entities. A trusted external entity includes a user with a valid PIN or other implementation specific applications, for example through network management applications.

7.2.5.5.22.2 If Bluetooth is supported and there is no record of the MHD in a DLNA device class's Bluetooth device database, then the device may create a temporary link key and distribute it to the MHD. This temporary link key is used for authentication and deriving the encryption key.

O	A	n/a	n/a	M-NCF	[2] [3]

7.2.5.5.23 NC M-NCF – BT PIN type

If Bluetooth is supported, then an M-NCF should implement support for variable PIN codes.

S	L	n/a	n/a	M-NCF	[4]

M-NCF is allowed to support fixed BT PIN type, but it is recommended to support variable PIN to enhance security.

7.2.5.5.24 NC M-NCF – Bluetooth PS support

7.2.5.5.24.1 If Bluetooth is supported, then an M-NCF must support the “sniff” Bluetooth power savings mode.

M	R	n/a	n/a	M-NCF	[4]

7.2.5.5.24.2 If Bluetooth is supported, then an M-NCF must accept at any time all MHD's requests related to setting the Bluetooth link between them in “sniff” mode.

M	L	n/a	n/a	M-NCF	[4]

7.2.5.5.25 NC M-NCF - NC-PS mode transitions with Bluetooth

7.2.5.5.25.1 If Bluetooth is supported, then an M-NCF must decide that the transition from the “active” to “standby” NC-PS mode is completed when the Bluetooth process to put the link from “active” to “sniff” Bluetooth mode is completed successfully.

M	A	n/a	n/a	M-NCF	[4]

While the NC-PS mode is “standby”, the MHD may request at any time to change the parameters of the “sniff” Bluetooth power saving mode. This does not trigger a transition out of the “standby” NC-PS mode.

The moment the M-NCF decides that the transition from “active” to “standby” NC-PS mode is completed, it must start any traffic reduction operations as described in 7.2.5.5.9.

7.2.5.5.25.2 If Bluetooth is supported, then an M-NCF must decide that the transition from the “standby” to “active” NC-PS mode is completed when the Bluetooth process to put the link from “sniff” to “active” mode is completed successfully.

M	A	n/a	n/a	M-NCF	[4]

The moment the M-NCF decides that the transition from “standby” to “active” NC-PS mode is completed, it must cease any traffic reduction operations as described in 7.2.5.5.9.

7.2.5.5.26 NC M-NCF - BT-Ethernet DLNAQOS access category mapping

If DLNAQOS is supported and an M-NCF bridges Bluetooth in the mobile connectivity domain and Ethernet in the home connectivity domain, then the device should maintain unmodified the IEEE 802.1Q headers and DSCP tags, when traffic is received from one connectivity domain and transmitted to the other.

S	A	n/a	n/a	M-NCF	[3]

7.2.5.5.27 NC M-NCF – BT-802.11 DLNAQOS access category mapping

The BNEP protocol is designed to encapsulate Ethernet packets over BT. Therefore, the BT priority taggings are the same as in Ethernet.

7.2.5.5.27.1 If DLNAQOS is supported and an M-NCF bridges Bluetooth in the mobile connectivity domain and 802.11 in the home connectivity domain, then packets received on the 802.11 interface and transmitted on the Bluetooth interface should include the IEEE 802.1D user priority value in IEEE 802.1Q header and the DSCP tag corresponding to the WMM access category of the received 802.11 packets in accordance with

Table 9 – BT-802.11 DLNAQOS access category mapping

WMM access category	IEEE 802.1D priority		DSCP
AC_BK	1	BK	0x08
AC_BE	0	BE	0x00
AC_VI	5	VI	0x28
AC_VO	7	NC	0x38

7.2.5.5.27.2 If DLNAQOS is supported and an M-NCF bridges Bluetooth in the mobile connectivity domain and 802.11 in the home connectivity domain, then packets received on the Bluetooth interface and transmitted on the 802.11 interface should include the WMM access category corresponding to the IEEE 802.1D user priority value in IEEE 802.1Q header tag of the received 802.3 packets in accordance with

Table 10 – IEEE 802.1D user priority values

IEEE 802.1D priority		DSCP	WMM access category
1	BK	0x08	AC_BK
2	-	0x10	
0	BE	0x00	AC_BE
3	EE	0x18	
4	CL	0x20	AC_VI
5	VI	0x28	
6	VO	0x30	AC_VO
7	NC	0x38	

S	A	n/a	n/a	M-NCF	IEEE 802.1D, IEEE 802.1Q [3] [10] [38]

7.2.5.5.27.3 If DLNAQOS is supported and an M-NCF bridges Bluetooth in the mobile connectivity domain and 802.11 in the home connectivity domain, if a packet received on the Bluetooth interface does not contain an 802.1Q tag, the M-NCF should look at the DSCP tag and map that to a WMM access category in accordance with Table 10 and preserve the DSCP tag across Bluetooth and 802.11 segments.

S	A	n/a	n/a	M-NCF	[3] [6] [10] [38]

7.2.5.5.27.4 If DLNAQOS is supported and an M-NCF bridges Bluetooth in the mobile connectivity domain and 802.11 in the home connectivity domain, if a packet received on the Bluetooth interface contains neither an 802.1Q tag or a DSCP tag, the packet should be passed through to the 802.11 interface without the addition of any priority tagging.

S	A	n/a	n/a	M-NCF	IEEE 802.1D, [3] [10] [38]

7.2.5.5.27.5 If DLNAQOS is supported and an M-NCF bridges Bluetooth in the mobile connectivity domain and 802.11 in the home connectivity domain, and a packet received on the 802.11 interface is not tagged with a WMM access category, then the packet should be passed through to the Bluetooth interface without the addition of any priority tagging.

S	A	n/a	n/a	M-NCF	IEEE 802.1D [3] [10] [38]

7.3 Device discovery and control

7.3.1 General

Subclause 7.3 covers the guidelines for implementing device discovery and control using the UPnP device architecture. These guidelines balance the needs for both devices and control points, and specify rules for a variety of protocol areas, such as SSDP, GENA events, SOAP actions, and HTTP transports for UPnP communications. It should be noted that HTTP guidelines in 7.3 apply only to UPnP-related transactions and not to content transfer transactions.

In 7.3, the following terms are used:

- UPnP endpoints: Refers to both UPnP devices and UPnP control points;
- HTTP clients: Refers to the HTTP clients used for UPnP communications. HTTP client guidelines in 7.3 do not apply to HTTP transport for content transfers or playback;
- HTTP servers: Refers to the HTTP servers used for UPnP communications. HTTP server guidelines in 7.3 do not apply to HTTP transport for content transfers or playback.

7.3.2 DDC UPnP device architecture

7.3.2.1 DLNA device classes and device capabilities must fully support the applicable mandatory portions of the UPnP device architecture v1.0 (UPnP DA) for discovery, description, control, eventing, and presentation.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

DLNA specifies UPnP device architecture 1.0 (UPnP DA) as the basic protocol framework for device classes.

7.3.2.2 A UPnP control point designed for a version of a UPnP device architecture must also be able to interoperate with later versions of the UPnP device architecture that have the same major version.

"Interoperate" means that a control point that has certain capabilities for older devices can at least provide the same capabilities for newer devices. For example, a control point that can discover an older UPnP device, parse its device and service description files, and invoke its UPnP actions must be able to do those same things with a UPnP device with a newer minor revision of the UPnP device architecture.

M	C	DMP DMC M-DMP +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMC M-DMU, M- DMD	MIU	[53]

Section 1 of the UPnP device architecture 1.0 indicates that advances in minor version of the UPnP device architecture are a superset of earlier (minor) versions with the same major version. This means that future UPnP devices with a newer minor revision will implement all of the behaviour required of the previous device architectures.

Although not explicitly stated by the UPnP device architecture, the intent of such backwards compatibility rules is to enable forward compatibility of control points with newer, minor revisions of the UPnP device architecture. Subclauses 7.3.2.2 and 7.3.2.3 formally require forward compatibility of control points, which is necessary for future interoperability.

Note that a version of the UPnP device architecture appears in the <specVersion> element of the device and service descriptions and the server header in SSDP, SOAP, and GENA messages.

One way to implement 7.3.2.2 is for the UPnP control point to treat the minor version a UPnP device architecture as 0 (i.e. ignore the minor version).

7.3.2.3 A UPnP control point designed for a version of a UPnP device type or service type must be able to interoperate with later versions of the same device type or service type.

"Interoperate" means that a control point that has certain capabilities for an older device can at least provide the same capabilities for a newer device of the same type and services. For example, a control point that can discover an older UPnP AV MediaServer and invoke its CDS:Browse action must be able to discover and invoke CDS:Browse on a newer UPnP MediaServer. The newer UPnP device may be newer because the device type and/or one of its associated UPnP services are of a newer version.

M	C	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

Section 2.1 of the UPnP device architecture 1.0 states that standardized device types and service types are required to be a superset of all previous versions of the same device/service type. This means that future UPnP device and service types will require all of the behaviour defined for previous versions.

It should be noted that a device version appears as part of the value in a <deviceType> element of a device description file. Similarly, the service version appears as part of the value in a <serviceType> element of a service description file. Version numbers also appear in NT, ST, and USN headers of SSDP messages. Furthermore, a service version appears in the xmlns namespace attributes of SOAP messages.

One way to implement 7.3.4.1 is for the UPnP control point to treat any version for the UPnP device and service types as a "1" (i.e., ignore the version number). Similarly UPnP devices and services can ignore the version numbers when comparing device type strings and service type string to maximize SSDP/MSEARCH and SOAP interoperability.

Lastly older control points that interact with newer UPnP device/service types are not expected to interoperate with conventions established for the newer device or service type, but they will still use parts of the UPnP device/service that are compatible with the older conventions.

7.3.2.4 If a SOAP action was defined in the specification of a previous service version, a UPnP control point may specify the xmlns namespace attribute for the service type and the SOAPACTION header in the SOAP request with the earlier service version.

O	C	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

In other words, if a SOAP action was defined in the specification of a previous service version (for example, version 1), then a UPnP control point may invoke the SOAP action with the earlier service version regardless of the service version described in the device description.

7.3.3 DDC UPnP Auto-IP support

7.3.3.1 UPnP devices and control points must fully support the Auto-IP specification (see [13]) even if they implement a DHCP server.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[13] [53]

DLNA device classes that do not properly support DHCP and AutoIP as required by the UPnP DA can cause IP addressing problems for other UPnP entities.

7.3.3.2 Whenever a UPnP device switches to a new IP address (whether assigned through Auto-IP or DHCP), the device should send an ssdp:byebye message for (and on) the old IP address.

"For (and on) the old IP address" means that the IP address indicated in the (UDP header of the) ssdp:byebye matches the old IP address of the UPnP device.

S	L	DMS DMR DMPr	M-DMS	n/a	[53]

This allows control points that discovered the UPnP device on the old IP address to know that the UPnP device is no longer available at the old address. However, this behaviour is not always possible for implementations built on some platforms.

7.3.3.3 If UPnP devices and control points use a self-assigned IP address, then they must implement duplicate address detection before assigning themselves the address.

M	R	HND +DN+ +PU+ +UP+ +PR1+ +PR2+	MHD	MIU	[53]

This guideline repeats a UPnP DA requirement that prevents the assignment of conflicting IP addresses.

7.3.3.4 If a DLNA device class implements a DHCP server, it must provide a mechanism to disable and enable the DHCP server.

M	A	HND	MHD	MIU	[53]

This guideline clarifies the condition for DHCP server support in a DLNA device. The user must be able to disable the DHCP server function to avoid the presence of multiple DHCP servers providing different network configurations on the same home network.

7.3.4 DDC UPnP SSDP default port

7.3.4.1 UPnP devices and control points must receive and process multicast discovery messages on port 1900.

M	C	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

These requirements ensure that UPnP endpoints will be more likely to discover each other on the home network.

7.3.4.2 UPnP devices must always explicitly specify port 1900 in every HOST header tag for every SSDP message.

M	R	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.4.3 UPnP control points receiving an SSDP message without the port number in the HOST header tag, must infer the port number is 1900.

M	R	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.4.4 UPnP control points must send MSEARCH messages using a source port greater than 1024 and not 1900.

M	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

These guidelines are based on a Microsoft technical advisory regarding security concerns for UPnP.

7.3.4.5 UPnP devices may ignore MSEARCH messages if the originating source port is less than, or equal to, 1024, or equal to, 1900.

O	L	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.5 DDC UPnP discovery robustness

7.3.5.1 UPnP endpoints (devices and control points) should wait a random amount of time, between 0 ms and 100 ms after acquiring a new IP address, before sending advertisements or initiating searches on a new IP interface.

S	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

This suggestion avoids SSDP discovery flooding on home networks that contain a large number of UPnP endpoints.

7.3.5.2 UPnP network devices must not send more than 10 ssdp:alive messages on a single network interface in any given 200 ms period.

M	L	DMS	DMR	DMP _r	M-DMS	n/a	[53]

This guideline prevents lost packets caused by buffer overflow of Ethernet drivers by UPnP devices with many services or embedded devices in the device hierarchy.

7.3.5.3 UPnP devices must send each advertisement set more than once on a single network interface. (It is recommended that UPnP devices send a total of 2 or 3 advertisement sets.)

An advertisement set refers to the set of 3+2d+k ssdp:alive messages that UPnP device sends as part of its periodic advertisements.

The repeated advertisement sets are referred to as duplicate sets.

The transmission windows for advertisement sets and duplicate sets cannot overlap in time.

M	C	DMS	DMR	DMP _r	M-DMS	n/a	[53]

This guideline clarifies how a UPnP device needs to retransmit its advertisements. However, implementers should remember that advertising too frequently runs the risk of flooding the SSDP channel.

7.3.5.4 A UPnP device that uses the same UD_N on multiple network interfaces, must send each individual ssdp:alive message (from an advertisement set) on all interfaces within a 10 second transmission window.

Time intervals between individual ssdp:alive messages on a single interface are not restricted by this requirement.

M	C	DMS	DMR	DMP _r	M-DMS	n/a	[53]

Control points need a way to determine the most reliable network route to the UPnP device. This guideline ensures that control points will receive an individual ssdp:alive message on all network interfaces within a 10 s transmission window.

7.3.5.5 The interval of sending these *advertisement groups* on a single network interface must be less than half the CACHE-CONTROL value.

The first advertisement set and the duplicate sets (transmitted on a single network interface) make up an advertisement group.

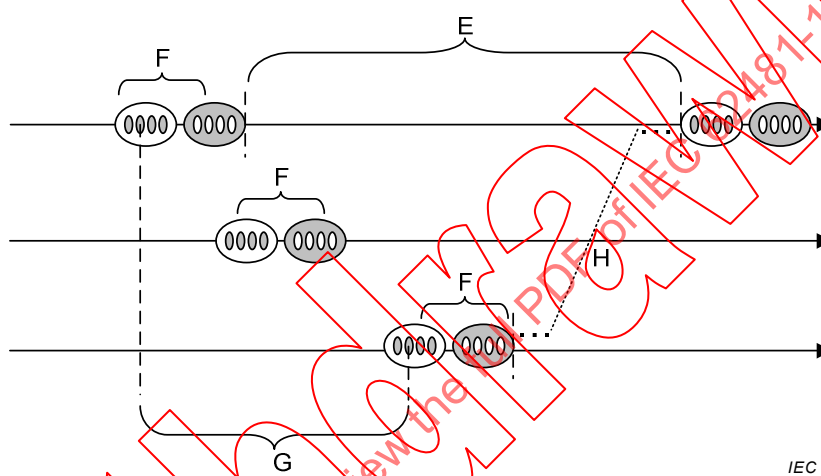
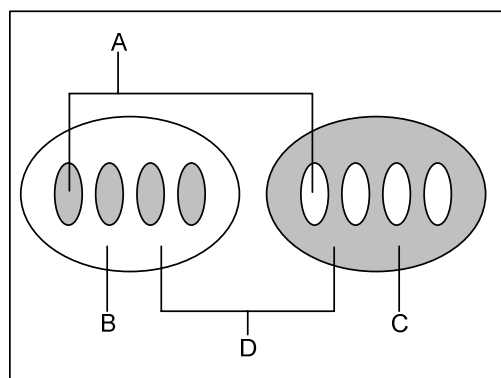
M	C	DMS	DMR	DMP _r	M-DMS	n/a	[53]

For consistency and interoperability, devices need to advertise more often than their notification cycle. However, implementers should remember that advertising too frequently runs the risk of flooding the SSDP channel.

7.3.5.6 The CACHE-CONTROL value should be at least 1800, as recommended in the UPnP device architecture.

S	R	DMS	DMR	DMP _r	M-DMS	n/a	[53]

Most devices that remain on the network for long periods should have CACHE-CONTROL value of 1800. However, some devices (mobile, wireless, etc.) that may want a smaller CACHE-CONTROL value.



IEC 1584/07

Key

- A One or more ssdp:alive messages, within advertisement sets and duplicate sets.
- B Advertisement set of $3+2d+k$ ssdp:alive messages.
- C Duplicate set of $3+2d+k$ ssdp:alive messages (see 7.3.5.3)
- D Combined advertisement set and duplicate sets make an advertisement group (see 7.3.5.5)
- E Delay between advertisement groups on same network is less than half of CACHE-CONTROL value (see 7.3.5.5)
- F Any arbitrary window of 200 ms have 10 or fewer ssdp:alive messages (see 7.3.5.2). An entire advertisement set need not fit inside the 200 ms window
- G An individual ssdp:alive message must have all corresponding ssdp:alive sent within a 10 s transmission window (see 7.3.5.4)
- H This delay is not drawn to scale

Figure 16 – UPnP discovery robustness

7.3.5.7 Due to the unreliable nature of UDP, control points should send each MSEARCH message more than once, not to exceed 10 MSEARCH requests in a 200 ms period. An MSEARCH message and its repeated duplicates should all be sent within a 10 s period.

S	R	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

Wireless access points do not retry multicast traffic and may cause UPnP discovery problems. This recommendation repeats advice from UPnP DA 1.0.1.

The 10 MSEARCH messages per 200 ms period is consistent with maximum saturation limit of 10 ssdp:alive messages per 200 ms period for UPnP devices (7.3.5.2). Likewise, the sending all of the MSEARCH messages in a window of 10 s is consistent with the requirement where all duplicates of an individual ssdp:alive message are sent within 10 s (7.3.5.4).

7.3.5.8 The control point should wait at least the amount of time specified in the MX header for responses to arrive from devices. The time waited for responses should be extended by additional time (a second or two) to allow for network propagation and processing delays.

S	R	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.5.9 Upon startup, UPnP devices should broadcast an ssdp:byebye before sending the initial ssdp:alive onto the local network.

S	L	DMS DMR DMPr	M-DMS	n/a	[53]

The UPnP device architecture specification does not account for devices that reset without sending an ssdp:byebye. If devices do not send an ssdp:byebye when returning to the network after such an event, control points cannot tell if the received announcement is for a new device instance, or is merely a periodic announcement for the same device instance.

Sending an ssdp:byebye as part of the normal start up process for a UPnP device ensures that UPnP control points with information about the previous device instance will safely discard state information about the previous device instance before communicating with the new device instance.

7.3.5.10 UPnP control points after acquiring a new IP address must initiate searches (for example, MSEARCH) on the new IP interface.

M	A	DMP DMC +UP+ +PR1+ +PR2+ +DN+ +PU+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

This guideline improves the user experience by allowing a UPnP control point (for example, DMP) to discover UPnP devices (for example, DMSs) more quickly after a UPnP control point and devices transition from a DHCP to an Auto-IP address or vice versa. Otherwise, a UPnP control point may have to wait until the next set of advertisements (i.e. NOTIFY packets) from a UPnP device to discover it, which could be a few minutes depending upon the CACHE-CONTROL max-age value. The timing of issuing MSEARCH is as specified in 7.3.5.1 as guidance.

7.3.6 DDC UPnP HTTP support and general rules

7.3.6.1 UPnP endpoints (devices and control points) must support at least HTTP/1.0 ([27]) for performing UPnP communications, excluding SSDP communications.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[27] [53]

This guideline specifies that UPnP transactions that use HTTP must minimally support HTTP/1.0.

7.3.6.2 UPnP devices must support HTTP/1.1.

M	R	DMS DMR DMPr	M-DMS	n/a	[39] [53]

Although HTTP/1.0 is the baseline for UPnP communications, HTTP/1.1 is recommended.

7.3.6.3 HTTP servers of UPnP control points must support HTTP/1.1.

M	R	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[39] [53]

7.3.6.4 HTTP clients of UPnP control points should use and support HTTP/1.1.

S	C	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[39] [53]

7.3.6.5 The message format of HTTP responses (sent by HTTP servers of both devices and control points) must be compliant with the version number specified by the request.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[29]

The clarifying IETF specification [29] states that HTTP/1.1 servers should return HTTP/1.1 even if the HTTP server receives a request marked with HTTP/1.0. The robustness rules, specified by the HTTP specification, enables clients and servers that employ different HTTP version numbers to coexist properly.

7.3.6.6 HTTP/1.1 servers of UPnP endpoints (devices and control points) should return HTTP version 1.1 in the response header, regardless of the version specified in the HTTP client's request.

S	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[29]

7.3.6.7 HTTP servers of UPnP endpoints (devices and control points) must not report a higher version of HTTP than is actually supported by the implementation.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

7.3.6.8 The HTTP servers and clients of UPnP endpoints (devices and control points) must be able to properly parse all HTTP headers provided to them. In particular, they must support HTTP header tags in any order and accept the tag name in a case-insensitive manner and associated data in a case-sensitive manner. If a header tag is not recognized by a UPnP endpoint, it must ignore the header and continue parsing the packet.

This guideline applies to all HTTP headers, regardless of whether the DLNA guidelines define a BNF syntax for the HTTP header value. In other words, all endpoints must implement a "parse and interpret" or "parse and ignore" when parsing an HTTP header field.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[27] [39]

This guideline specifies a minimal robustness level for parsing HTTP headers. The HTTP headers include both HTTP headers defined in [39] and other headers, such as DLNA defined and vendor defined headers, used for DLNA interoperability. Under no circumstances should the UPnP endpoint fail to process a properly formed HTTP packet because of an unrecognized or unsupported field.

7.3.6.9 The HTTP servers and clients of UPnP endpoints (devices and control points) must include the content-type header tag in every UPnP-related TCP-based HTTP transaction (SOAP, GENA, and device/service description) that contains an XML body. This content type must always be marked as the following:

- text/xml; charset="utf-8"

It should be noted that charset parameter value is case-insensitive and double quotations may be omitted.

Furthermore, the XML must be encoded in UTF-8.

UPnP endpoints (devices and control points) that receive a content type of text/xml must infer UTF-8 character set encoding.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[27] [39]

Restricting UPnP communications to UTF-8 simplifies implementations and makes it so that devices need not implement a separate parsing engine for every local region.

7.3.6.10 If the DLNA guidelines define a BNF syntax for an HTTP header, then the HTTP servers and clients of UPnP endpoints must not include white spaces in the header-value of HTTP headers unless SP and LWS are explicitly specified in the syntax (BNF) definitions.

If the DLNA guidelines do not define a BNF syntax for an HTTP header, then the header must conform to the message-header syntax in 4.2 of [39], regardless of whether the HTTP header is defined in [39] or if the HTTP header is vendor-defined. It should be noted that the syntax for field-value permits LWS to separate tokens and other data in the field-value.

Implied LWS between the HTTP header-name and the HTTP header-value are permitted as specified in [39], regardless of whether the DLNA guidelines specify a BNF syntax for the HTTP header.

M	A	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

Conformance to 6.4 of [39], the restriction on the use of SP and LWS characters is applied in UPnP HTTP communications. It should be noted that it is still allowed to include white spaces between header-name and header-value.

The header-name and header-value are defined by the field-name and field-value tokens of the message-header syntax in 4.2 of [39].

7.3.7 DDC UPnP HTTP/1.0 rules

7.3.7.1 For all HTTP/1.0 transactions, the HTTP server must immediately close the TCP connection after sending the complete HTTP response. This guideline covers both kinds of HTTP/1.0 transactions:

- HTTP/1.1 server responds to an HTTP/1.0 request; and
- HTTP/1.0 server responds to an HTTP/1.1 request.

M	C	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[27] [39]

The use of the content-length field greatly reduces the parsing complexity of HTTP message bodies on a UPnP control point.

When an HTTP server responds to an HTTP/1.0 request without closing the socket, the content-length field is the only method that the client can use to determine that the entire response was received.

7.3.7.2 If a UPnP device's HTTP server responds to a SOAP request as part of an HTTP/1.0 transaction, then the UPnP device must close the TCP connection after the response has been sent.

This guideline covers both kinds of HTTP/1.0 transactions:

- HTTP/1.1 server responds to an HTTP/1.0 request; and
- HTTP/1.0 server responds to an HTTP/1.1 request.

It should be noted that in both of the above cases, the UPnP device has the HTTP server and the UPnP control point issues the HTTP request.

M	R	DMS DMR DMPr	M-DMS	n/a	[27] [39]

This is the proper behaviour for a UPnP device, as it follows standard HTTP rules.

7.3.7.3 If a UPnP control point's HTTP server responds to a GENA event as part of an HTTP/1.0 transaction, then the control point must close the TCP connection after the response has been sent.

This guideline covers both kinds of HTTP/1.0 transactions:

- HTTP/1.1 server responds to an HTTP/1.0 request; and
- HTTP/1.0 server responds to an HTTP/1.1 request.

It should be noted that in both of the above cases, the UPnP control point has HTTP server and the UPnP device issues the HTTP request.

M	R	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[27] [39]

This is the proper behaviour for a UPnP control point.

7.3.8 DDC UPnP HTTP/1.1 transaction rules

7.3.8.1 A UPnP device's HTTP server must close the TCP connection after responding to a SOAP request with the CONNECTION:CLOSE token.

M	R	DMS DMR DMPr	M-DMS	n/a	[39]

7.3.8.2 A UPnP control point's HTTP server must close the TCP connection after responding to an event that was sent with the CONNECTION:CLOSE token.

M	R	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[39]

7.3.8.3 HTTP clients of UPnP endpoints (devices and control points) must not report support for HTTP/1.1 unless they also support chunked transfer coding and correctly parse a 100 (continue response), as required by the HTTP/1.1 specification.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

Only HTTP clients that support chunked transfer coding and 100 (continue response) messages can initiate HTTP/1.1 transactions.

7.3.8.4 The HTTP servers of UPnP endpoints (devices and control points) must use the content-length HTTP header tag at all times, unless the connection will be closed immediately after the response is sent or chunked transfer coding is used.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

When an HTTP/1.1 server sends a response back to the client without closing the socket afterwards, the client will not know when the entire response was received, unless the response was encoded with chunked transfer coding, without interpreting the content-length header.

7.3.8.5 The HTTP clients of UPnP endpoints (devices and control points) may issue HTTP/1.1 requests encoded with chunked transfer coding.

O	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

These guidelines repeat the HTTP specification by noting the permitted use of chunked transfer coding for HTTP/1.1 requests. HTTP clients of UPnP devices can use chunked transfer coding for delivery of UPnP GENA events. HTTP clients of UPnP control points can use chunked transfer coding for delivery of UPnP SOAP actions. As such, HTTP/1.1 servers of UPnP endpoints are required to support HTTP/1.1 requests encoded with chunked transfer coding.

7.3.8.6 The HTTP servers of UPnP endpoints (devices and control points) must accept, decode, and respond to HTTP/1.1 requests encoded with chunked transfer coding.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

7.3.9 DDC UPnP HTTP persistent connections

7.3.9.1 The HTTP clients and servers of UPnP endpoints (devices and control points) must not use the CONNECTION:KEEPALIVE token for HTTP/1.0 transactions.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[27] [39]

Although HTTP/1.0 supports persistent connections by way of the KeepAlive token, there are interoperability issues because the methodology is an experimental extension.

7.3.9.2 The HTTP clients and servers of UPnP endpoints (devices and control points) should support persistent HTTP/1.1 connections and pipelining.

S	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

Persistent HTTP connections allow devices and control points to use fewer resources when communicating. Pipelining adds the ability for control points to queue requests onto an existing session.

7.3.9.3 The HTTP clients of UPnP endpoints should use persistent HTTP/1.1 connections.

S	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

It should be noted that the default behaviour for HTTP/1.1 is a persistent connection. Persistent connections result in no accumulation of TCP TIME-WAIT because the originator of the connection closes the socket.

7.3.9.4 The HTTP clients of UPnP endpoints must fall back to non-pipelining if the connection is closed after the first request and a second (or more) request from the same network entity is pending.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

This guideline ensures consistent and correct behaviour between mixes of UPnP endpoints that may or may not support HTTP pipelining.

7.3.9.5 The HTTP servers of UPnP endpoints (devices and control points) that do not support persistent connections must answer the first HTTP request from the requesting UPnP control point and close the TCP connection to correctly ignore other requests.

M	C	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

7.3.9.6 The HTTP clients of UPnP endpoints that send multiple requests in a single HTTP session must be ready to open new HTTP sessions if the device does not respond to all requests on the initial HTTP session.

M	C	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

7.3.9.7 The HTTP clients of UPnP endpoints must close a persistent connection (HTTP/1.1) within 60 s of inactivity (i.e. no traffic and no pending requests).

This guideline applies to both UPnP devices and control points. Context of this guideline is specific to UPnP-related communications, excluding SSDP communications. This guideline does not apply to the transport layer communications for media content transfers.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

This prevents control points and devices from holding network sockets for an unnecessarily long period.

7.3.10 DDC UPnP device responsiveness

7.3.10.1 UPnP device architecture specification requires UPnP devices to complete the SOAP response in 30 s. However, this can be difficult to guarantee at the implementation layer for all types of UPnP actions. These guidelines attempt to strike a balance between ideal goals and practical implementation needs for both devices and control points.

That being stated, the original inspiration for these guidelines is that some UPnP AV MediaServer devices cannot guarantee that a response will complete within 30 s for a variety of reasons. Network bandwidth, query complexity, and hardware performance can vary. This being the case, such devices must still begin their response within 30 s.

It should be noted that a UPnP AV MediaServer can reduce a long transmission time for a SOAP response (for a CDS:Browse or CDS:Search action) by reducing the number of returned items in the result. See 7.4.1.4.20 for more information.

7.3.10.2 UPnP devices must begin the transmission of a SOAP response within 27 s of receiving a complete SOAP request.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.10.3 UPnP devices should begin the transmission of SOAP responses as soon as possible.

S	C	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.10.4 UPnP devices should complete the transmission of a SOAP response within 29 s.

S	C	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.10.5 A UPnP control point may terminate the TCP connection for a SOAP response transmission that exceeds 30 s.

O	C	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU M-DMD	MIU	[53]

7.3.11 DDC UPnP device description rules

7.3.11.1 The total byte size of a device description file must not exceed 20 480 bytes (20 KB). This byte limit includes the HTTP headers.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

Provides a known maximum size for device description documents.

7.3.11.2 DLNA UPnP devices must employ the <dlina:X_DLNADOC> XML element inside the <device> element of the device description document to indicate adherence to a particular DLNA home networked device interoperability guidelines document version. The value of this element is the DLNA device class, a dash character, followed by the numeric version value of the interoperability guidelines document.

The <dlina:X_DLNADOC> element indicates DLNA compliance for a specific <device>, excluding its embedded devices listed in <deviceList>.

The value of the <dlina:X_DLNADOC> element is a string as defined below. Linear white spaces (LWS) are not implied in this definition.

- dlndoc-value = dlndev-class "-" dlndev-version;
- dlndev-class = "DMS" | "DMR" | "DMP" | "M-DMS" | other-dev-class;
- other-dev-class = *("<A" – "Z", "a" – "z", "-");
- dlndev-version = major-version "." minor-version;
- major-version = DIGIT;
- minor-version = DIGIT DIGIT.

The dlndev-class represents a device class of a DLNA device.

The dlndev-version represents a version of the interoperability guidelines supported by the DLNA device class.

An example of <dlndev:X_DLNDDOC> element is shown as follows:

```
<dlndev:X_DLNDDOC xmlns:dlndev="urn:schemas-dlna-org:device-1-0">DMS-1.50</dlndev:X_DLNDDOC>
```

M	A	DMS DMR DMP	M-DMS	n/a	[53]

Provides an easy way of distinguishing UPnP devices that are claimed as being DLNA devices.

This guideline specifies the scoping rules for the <dlndev:X_DLNDDOC> element. Essentially, UPnP devices (in a device hierarchy) must be marked explicitly as being DLNA devices. Although the subject matter is technically outside the scope, this guideline permits a non DLNA UPnP device to be listed in a device hierarchy that has DLNA devices.

The<dlndev:X_DLNDDOC> element may appear multiple times such as the case for a DMS and M-DMS combination device.

7.3.11.3 The<dlndev:X_DLNDDOC> element may appear multiple times.

O	A	DMS DMR DMP	M-DMS	n/a	[53]

7.3.11.4 UPnP control points must be matched against multiple <dlndev:X_DLNDDOC> elements. Specifically, a control point that claims to discover a particular type DLNA device class must be able to discover that type of DLNA device class, even if the specific <dlndev:X_DLNDDOC> element of interest is not the first <dlndev:X_DLNDDOC> in the device description document.

M	A	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M-DMU, M-DMD	MIU	[53]

7.3.11.5 The namespace "urn:schemas-dlna-org:device-1-0" must be specified in the <root> element or the <dlndev:X_DLNDDOC> element and the namespace prefix must be "dlndev:".

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.11.6 UPnP control points must ignore the element value of <dlna:X_DLNADOC>. For example, DLNA control points must not filter out a DLNA device because the version value is different from that expected.

M	A	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

In the near-term, the <dlna:X_DLNADOC> version number is useful for testing purposes. Future guidelines will specify behaviour for interoperability between newer and older DLNA devices and the purpose of this field may change.

7.3.11.7 If a vendor builds an implementation of a device class (with zero or more device capabilities or device options), then the implementation must comply with all mandatory portions of the interoperability guidelines for the specified dlna-version token 7.3.11.2. If a vendor implements an older version of the interoperability guidelines, then the vendor must not implement guidelines defined in newer versions of the interoperability guidelines.

M	C	HND	MHD	MIU	[53]

Vendors are not permitted to selectively implement portions of newer versions of the interoperability guidelines.

For example, if a vendor wants to build a DMS that supports RTP in the DLNA-defined manner, then the vendor must implement the DMS according to the 1.50 (or newer) version of the interoperability guidelines, which includes using a value of "1.50" for the dlna-version token. A DMS implementation that uses a "1.00" value for the dlna-version but implements guidelines that are specific to newer versions of the interoperability guidelines demonstrates a violation of this guideline.

The primary reason for this guideline is to ensure that newer implementations participate in the DLNA networking ecosystem in a manner that is consistent with the assumptions of those guidelines. Newer versions of the interoperability guidelines are drafted with compatibility for previous versions, but newer versions sometimes have new mandatory requirements. The new mandatory guidelines often ensure the robustness of the network, which is needed when the interoperability guidelines increase network complexity through new usages.

This guideline makes no claim on policy decisions about granting a DLNA logo/certification to implementations that use older versions of the interoperability guidelines. DLNA reserves the right to require a baseline version of the interoperability guidelines for future implementations.

7.3.12 DDC UPnP embedded device support

7.3.12.1 DLNA UPnP devices must not have more than 6 total UPnP devices in the device hierarchy with a maximum depth of 4. Root devices have a depth of 1.

M	L	DMS DMR DMP _r	M-DMS	n/a	[53]

A UPnP control point must handle DLNA devices that include a combination or an aggregate of devices and services. A specific limit sets a bound on memory and processing requirements for control points.

7.3.12.2 UPnP control points must support device hierarchies that have up to a total of 6 DLNA devices with a maximum depth of 4. Root devices have a depth of 1.

M	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.12.3 DLNA UPnP devices must be functionally independent even if they are in the same device hierarchy. In other words,

- a DLNA UPnP device is identified as a <device> with the <dlina:X_DLNA DOC> element;
- a DLNA UPnP device has no functional dependency with other UPnP devices in the device hierarchy;
- a DLNA UPnP device has no functional dependency with other DLNA UPnP devices in the device hierarchy.

A DLNA UPnP device (Device-A) is functionally independent if it does not require a control point to invoke a UPnP action of another UPnP device (Device-B) in order to put Device-A in a state for use with a DLNA compliant UPnP control point. It should be noted that the definition assumes Device-A and Device-B are in the same device hierarchy. Furthermore, Device-B may or may not be a DLNA device, as indicated by the presence of the <dlina:X_DLNA DOC> element.

M	L	DMS DMR DMP _r	M-DMS	n/a	[53]

These guidelines simplify control point implementations by not requiring them to know about any functional dependencies between DLNA UPnP devices found in a device hierarchy.

Although the subject-matter is technically outside the scope, this guideline does not prohibit the use of a UPnP device that has functional dependence on another UPnP device. However, a device that has a functional dependence cannot be marked with a <dlina:X_DLNA DOC> element.

7.3.12.4 DLNA control points must not assume any functional dependency between embedded devices that contain the <dlina:X_DLNA DOC> element.

For example, a control point that requires UPnP actions to be called on one of the UPnP devices before calling UPnP actions on another UPnP device (both UPnP devices belong to the same hierarchy and have the <dlina:X_DLNA DOC> element) is in violation of this requirement.

M	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.12.5 UPnP devices may be implemented as a descendent of a UPnP root device, which may or may not be a standard UPnP device type.

O	C	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.12.6 UPnP control points must interoperate with embedded DLNA devices that exist in device hierarchies where the root happens to be a non-standard UPnP device type (i.e. vendor-defined UPnP device type).

M	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.12.7 UPnP devices that are not DLNA-compliant may be listed in a device hierarchy.

These non-DLNA UPnP devices count against the maximum number of six total UPnP devices in the device hierarchy, indicated in 7.3.12.1

O	L	DMS DMR DMPr	M-DMS	n/a	[53]

This guideline permits that a device hierarchy to have UPnP devices that do not have the <lna:X_DLNADOC> element.

7.3.13 DDC UPnP service description rules

7.3.13.1 Optional actions listed in the SCPD must be supported and not return the NOT_IMPLEMENTED UPnP error in response to an invocation. Optional actions that are not implemented must not be listed in the SCPD.

M	C	DMS DMR DMPr	M-DMS	n/a	[53]

UPnP devices must fully and accurately reflect capabilities required by the standardized device control protocol (DCP) and listed in the UPnP device's service control protocol document (SCPD).

7.3.13.2 A UPnP state variable must not be present unless it meets at least one of the following characteristics:

- the UPnP state variable is actually used by the device, either as an evented state variable or as an action parameter;
- the UPnP state variable is normatively defined by a UPnP service to neither be evented nor used for an action parameter.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

This guideline is in consideration of the fact that control points may run on a platform with limited resources.

7.3.13.3 If an allowed value list or value range is specified, UPnP devices should accept all values in the state variable range, regardless of the stepping (as indicated by a <step> element of the UPnP state variable).

S	L	DMS DMR DMPr	M-DMS	n/a	[53]

Although control points should employ logic for correctly checking an argument for compliance against a device's stepping, this is not always the case. For broader interoperability, this guideline is suggested for UPnP devices but it is not mandatory. It should be noted that the AVTransport, ContentDirectory, and ConnectionManager services do not have state variables that use stepping by default.

7.3.13.4 Services with evented state variables must support SUBSCRIBE and UNSUBSCRIBE operations.

M	R	DMS DMR DMPr	M-DMS	n/a	[53]

Specifies normative behaviour for services with evented state variables.

7.3.13.5 DCP-required or SCPD-specified state variables with the attribute SendEvent="Yes" must actually be evented.

M	R	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.13.6 Service description files must not exceed 51 200 bytes (50 KB). This byte limit includes the HTTP headers.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

This provides a reasonable maximum length for service description files.

7.3.14 DDC UPnP XML namespace

Default namespace defined by the UPnP DA must be used in device and service descriptions.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

All standard elements in device and service descriptions do not use a namespace prefix.

7.3.15 DDC UPnP action argument encoding

UPnP devices must accept SOAP action arguments in the same order as specified in the standardized DCP.

The order of arguments of SOAP actions in an SCPD must match the standardized DCP.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.16 DDC UPnP SOAP packet size

7.3.16.1 UPnP devices must be able to accept SOAP requests that are up to 20 480 bytes (20 KB) in size. This byte limit includes the HTTP headers.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

This guideline provides control points with a minimal SOAP packet size (total size for headers and body). It is understood the support of larger SOAP requests is permitted.

7.3.16.2 UPnP control points must be able to accept SOAP responses that are up to 204 800 bytes (200 KB) in size. This byte limit includes the HTTP headers.

M	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

Security recommendations call out 200 KB as a reasonable upper bound for SOAP responses (total size for headers and body).

7.3.16.3 UPnP control points may refuse SOAP responses that are more than 204 800 bytes (200 KB) in size. This byte limit includes the HTTP headers.

Control points may implement the not accept SOAP response behaviour by terminating the TCP connection after 200 KB is reached.

O	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.17 DDC UPnP error codes

7.3.17.1 UPnP endpoints (devices and control points) should use and return the proper error code when encountering an error condition for a UPnP operation. This includes using the proper HTTP error codes and method error codes for UPnP actions. In some extreme circumstances, it may be necessary to simply close a UPnP initiated connection upon encountering an error condition.

S	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

This requirement covers the proper expected behaviour for any UPnP endpoint and is repeated here due to its importance in gracefully recovering from error conditions on a distributed home network.

UPnP devices should not use the UPnP error code value of 403 (Out of Sync), since that error code value is deprecated in the draft version of the UPnP DA v1.0.1.

7.3.17.2 UPnP control points must be able to tolerate unknown method error codes for UPnP actions

M	A	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

Ideally, UPnP control points should treat all unknown method error codes for UPnP actions as a generic error condition.

7.3.17.3 HTTP clients for UPnP endpoints (devices and control points) are not required to understand unknown HTTP status code values, but they must understand the class of the status code. The class of the status code is indicated by the first digit of the status code numeric value. HTTP clients must treat unrecognized status code values as equivalent to the x00 code of the class.

M	C	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

7.3.18 DDC UPnP GENA packet size

7.3.18.1 UPnP control points must be able to accept GENA event transmissions that are up to 20 480 bytes (20 KB) in size. This byte limit includes the HTTP headers.

M	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

This guideline specifies the minimum capability of control points to receive events of 20 KB in size (for headers and body). Control points are permitted to support larger GENA events.

7.3.18.2 UPnP control points may choose not to accept GENA event transmissions that are more than 20 KB in size.

Control points may implement the not accept GENA event behaviour by terminating the TCP connection after 20 KB is reached.

O	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.19 DDC UPnP subscription handling

7.3.19.1 The SUBSCRIBE response must include the Content-Length: 0 HTTP header/value pair, if the response is not encoded with Chunked Transfer Coding.

The only exception to this rule is if the device can guarantee a TCP 'FIN' packet is sent before the initial event message is sent to the subscribing control point.

M	L	DMS DMR DMPr	M-DMS	n/a	[53]

In order for a control point to receive the initial event from a UPnP device, a control point needs to know the Subscription ID (SID) value.

The SID is obtained in the response to a SUBSCRIBE request.

Therefore, a control point must receive the entire SUBSCRIBE response before it receives the first event.

The HTTP clients of control points only have two ways to know when the SUBSCRIBE response has finished. The first is to complete the transaction when the Content-Length:0 values are specified. The second is to receive the TCP "FIN" flag in the TCP stream.

7.3.19.2 UPnP devices must assign a globally unique SID, where the global context is defined as the UPnP network. The format for the uuid is as specified in 7.3.20.

M	C	DMS DMR DMPr	M-DMS	n/a	[53]

See 7.3.21 for a way to generate a globally unique SID.

7.3.20 DDC UPnP UUID format

The format of the SID is "uuid:" followed by a UUID, which is a 128-bit value represented in hexadecimal form, with optional hyphens throughout the encoding. The maximum length is 68 bytes, including the "uuid:" portion.

Example:

uuid:00000000-0000-0000-0000000000000000

M	C	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.21 DDC UPnP UUID generation

UPnP devices should use the DCE 1.1 methodology for generating a globally unique UUID value.

S	R	DMS DMR DMPr	M-DMS	n/a	[73]

There are several ways to generate a UUID value. The best ways to generate a UUID involve using some form of a network address and the current time, such as the algorithm described in [73].

7.3.22 DDC UPnP event subscription renewals

This guideline instructs developers that the control point is responsible for renewing subscriptions in a timely manner.

If UPnP control points want to continue receiving UPnP events, then they must renew their subscriptions before the negotiated subscription TIMEOUT expires.

M	C	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.23 DDC UPnP event notification handling

7.3.23.1 UPnP devices must send events to all properly subscribed UPnP control points. The device must enforce a subscription TIMEOUT value of 5 min.

The UPnP device behaviour of enforcing this 5-min TIMEOUT value is implemented by specifying "TIMEOUT: second-300" as an HTTP header/value pair.

M	R	DMS DMR DMPr	M-DMS	n/a	[53]

A UPnP control point that subscribes to events and then subsequently leaves the UPnP network will cause a UPnP device to possess an event subscription to an invalid address. The device must not hold up events to other subscribing control points while, for example, the HTTP session with the absent UPnP control point times out. This scenario has been a major cause of UPnP control point disruption and usability problems. UPnP control points that stop receiving events may incorrectly indicate to the user that a device is stalled or is malfunctioning.

7.3.23.2 UPnP devices should monitor their subscription lists and remove control points that fail to renew their subscription within the negotiated time.

S	R	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.24 DDC UPnP unknown header/tag/field robustness rule

UPnP endpoints (devices and control points) must be tolerant of unknown headers, tags, fields, attributes, and values for HTTP, SSDP, XML, SOAP, and GENA. Specifically, this tolerance guideline applies to

- HTTP headers, tokens, values;
- SSDP headers, tokens, values;
- unknown XML elements and attributes of SOAP or GENA fragments;
- unknown XML elements and attributes in device description files or service description files.

Tolerant behaviour is defined as being able to successfully "parse and interpret" or "parse and ignore" the unknown text.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[27] [39] [53]

This guideline addresses forward compatibility and also ensures broader interoperability between implementations that employ vendor extensions in the manner described by the guideline.

7.3.25 DDC URI rules

7.3.25.1 All absolute URIs used for UPnP communications must use IP addresses (not host names).

UPnP communications specifically refer to the following areas:

- SOAP actions;
- GENA events;
- device description files;
- service description (SCPD) files;
- UPnP presentation files;
- SSDP messages.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

These guidelines are mandatory because DLNA device classes cannot depend on DNS infrastructure within a home network environment.

7.3.25.2 The a.b.c.d format for Ipv4 addresses must be used for UPnP communications, where each quad represents a byte in network byte order form.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

7.3.25.3 HTTP URI escaping is always performed according to the URI specification ([25]) as required in 3.2.1 of the HTTP/1.1 specification [39].

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[25] [39]

This guideline specifies how to escape URI values.

7.3.25.4 All URIs used for UPnP communications must not exceed 256 bytes in URI-escaped UTF-8 encoded form. This guideline applies to both absolute URIs and complete URIs (relative URIs combined with a base path).

UPnP communications do not cover the URIs in the presentation files themselves, nor does it cover informational URIs for the manufacturer or product/model.

M	L	HND MIU, +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

These guidelines provide a maximum URI length for the UPnP layer. See 7.4.1.3.10 and 7.4.1.3.10.4 for the maximum URI length at the UPnP AV layer.

According to 2.10 of [64], white spaces are significant (i.e. non-markup characters) in XML elements that contain character data. Therefore, XML elements that contain a single (absolute or relative) URI value cannot have preceding or trailing white spaces.

7.3.25.5 All URIs (not used for UPnP communications) must not exceed 1 024 bytes, in the URI-escaped UTF-8 encoded form. This guideline covers URIs, such as (but not limited to):

- URIs inside UPnP presentation files;
- URIs in the device description for product, model, or manufacturer information.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

7.3.25.6 UPnP devices should not use the <URLBase> element in the device description document. When this element is omitted, the <URLBase> is the base path of the LOCATION value (URL) in the device advertisement.

S	L	DMS DMR DMPr	M-DMS	n/a	[53]

These requirements have several benefits. Since the device description and service description documents will no longer include IP addresses and port numbers, UPnP devices are simplified. The document can be sent even if the IP address changes or the device is multi-homed. UPnP control points will have an easier time dealing with UPnP devices that meet these requirements, and will be able to handle any situation that arises.

7.3.25.7 If a URI in a device description or SCPD (service description) file is used for SOAP actions, GENA events, SCPD files, or UPnP presentation files, then the URI may be a relative URI with its base as the base path of the LOCATION URL value.

The following is an example of the above guideline:

- the ssdp:alive message has LOCATION value of http://172.16.0.2/devicedesc.xml

The device description file does not have the <URLBase> element.

One of the services has these element values:

- <SCPDURL> has "/service_desc.xml";
- <controlURL> has "control";
- <eventSubURL> has "http://172.16.0.2:3000/sub".

The full URL for that service is as follows:

- SCPDURL: http://172.16.0.2/service_desc.xml;
- controlURL: http://172.16.0.2/control;
- eventSubURL: http://172.16.0.2:3000/sub.

O	R	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.25.8 UPnP control points must work with UPnP devices that use a <URLBase> element and with those that do not use a <URLBase> element. Control points must also work with UPnP devices that use absolute or relative URIs.

M	R	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.25.9 UPnP devices must use the CALLBACK URI value sent by control points for event delivery. (i.e., a UPnP device must use the CALLBACK URI value exactly as received that are consistent with 7.3.25.1 to 7.3.25.4).

M	R	DMS DMR DMPr	M-DMS	n/a	[53]

Callback URI requirements help ensure interoperability by limiting edge cases.

7.3.25.10 UPnP control points must not specify more than one CALLBACK URI value for the CALLBACK header in a request with the SUBSCRIBE method.

M	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

Simplifies a UPnP device's implementation for GENA eventing.

7.3.26 DDC UPnP device description usage

7.3.26.1 If a DLNA UPnP device wants to change a device description or a service description files, then the UPnP device must need to

- first leave the UPnP network by sending an ssdp:byebye message;
- then change the desired device description or service description files; and
- finally join the UPnP network with the new XML files using an ssdp:alive message.

M	C	DMS DMR DMPr	M-DMS	n/a	[53]

UPnP control points often bind UDN values to device representations. Therefore, a device that changes its logical representation causes problems if it uses the same UDN. This guideline does not apply if the device sends an ssdp:byebye message. In such cases, the device can still keep its UDN value and change its logical representation before rejoining the UPnP network.

7.3.26.2 A DLNA UPnP control point that removes a UPnP device from its list of active devices must also invalidate its local representation of the device.

The control point removes a device for a variety of reasons, such as a CACHE-CONTROL timeout.

Invalidating the local representation of the device means that the control point must reload the device description and service description files.

M	C	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

This guideline obligates a control point to refresh device description and service description documents the next time the device is discovered. This, for example, allows a device to add additional supported actions or services (via firmware update), without having to change the UDN of the device.

7.3.27 DDC UPnP UDN usage

7.3.27.1 UPnP devices should not change the UDN between reboots or application launch/shutdown.

S	A	DMS	DMR	DMP	M-DMS	n/a	[53]

Implementing this guideline enables usages such as my favourite devices. Generally, UPnP device UDN values should be long-lived.

UPnP DA states as follows;

UDN: Must be the same over time for a specific device instance (i.e. must survive reboots).

7.3.27.2 UPnP devices must not change the UDN if only the <friendlyName> or IP addresses values are changed.

M	A	DMS	DMR	DMP	M-DMS	n/a	[53]

UPnP control point can identify UPnP devices even if FriendlyName or IP addresses are changed.

7.3.27.3 In conjunction with the restrictions in 7.3.27.2, UDN may be changed if a UPnP device changes its device description or any of its supported services.

O	A	DMS	DMR	DMP	M-DMS	n/a	[53]

7.3.27.4 If a UPnP device UDN changes, it must re-advertise on the network using the new UDN.

M	C	DMS	DMR	DMP	M-DMS	n/a	[53]

Control points that receive the advertisement know that a new UPnP device is available. This is required by 7.3.11.

7.3.27.5 If a UPnP device UDN changes, it must send an ssdp:byebye for the old UDN.

M	C	DMS	DMR	DMP	M-DMS	n/a	[53]

Without this guideline, UPnP control points will have no idea that the old UDN is no longer valid. This is required by 7.3.11.

7.3.27.6 A UPnP device must limit their UDN to a UTF-8 encoded string value containing "uuid:" followed by a UUID as specified in 7.3.22.

M	C	DMS DMR DMPr	M-DMS	n/a	[53]

See 7.3.21 for a way to generate a globally unique 128-bit value for the UDN.

7.3.28 DDC UPnP multi homing rules

Multiple home network segments, wireless networking, and Auto-IP can combine to create usability problems that can be avoided by following the specified rules.

7.3.28.1 When a UPnP device has multiple IP addresses, the device may advertise on those IP addresses with the same or different UDN.

O	C	DMS DMR DMPr	M-DMS	n/a	[53]

Multiple home network segments, wireless networking, and Auto-IP can combine to create usability problems that can be avoided by following the specified rules.

7.3.28.2 Each UPnP device advertisement must contain a return IP address of the home network interface it is sent on.

M	C	DMS DMR DMPr	M-DMS	n/a	[53]

7.3.28.3 Upon receiving multiple advertisements for the same UPnP device UDN, a UPnP control point should select the vendor-defined preferred advertisement as the route to the device.

S	C	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.28.4 When a UPnP control point gets an advertisement for a UPnP device UDN on a different IP address from the one it has previously selected, it may continue to use its selected IP address provided that it has received an advertisement on the selected IP address in the last 10 s. Otherwise, if the UPnP control point does not receive an advertisement for its selected IP address in the next 10 s, it may change its selection to the new IP address. Even if the control point keeps the selected IP address in this case, it should change its selection to the new IP address when an access to the selected IP address fails.

O	L	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[53]

7.3.29 DDC UPnP device icons

7.3.29.1 If a UPnP device provides a device icon, the UPnP device must provide two JPEG icons that conform to the 7.1.6 JPEG SM ICO format profile and 7.1.7 JPEG LRG ICO format profile media format profiles and two PNG icons that conform to the 7.2.2 PNG SM ICO format profile and 7.2.3 PNG LRG ICO format profile media format profiles. These can be located in IEC 62481-2.

M	L	DMS DMR DMP _r	M-DMS	n/a	IEC 62481-2 [53]

This requirement will ensure device icon compatibility and good authoring practices. The reason for requiring PNG icons is that the lossless compression is much better for small size images. Furthermore, alpha-blending makes it possible to present better user interfaces.

7.3.29.2 UPnP devices may provide additional icons in other formats besides PNG and JPEG.

O	R	DMS DMR DMP _r	M-DMS	n/a	[53]

7.3.29.3 UPnP devices must use <mimetype>, <width>, <height>, <depth>, and <url> sub-elements for an <icon> element within its device description.

The value of <mimetype>, <width>, and <height> elements for DLNA device icons must conform to the DLNA icon media format profiles 7.1.6 JPEG SM ICO Format Profile, 7.1.7 JPEG LRG ICO format profile, 7.2.2 PNG SM ICO format profile and 7.2.3 PNG LRG ICO format profile, respectively. These can be found in IEC 62481-2.

Values in Table 11 are recommended for the <depth> element which indicates colour bits per pixel for PNG and JPEG device icons.

Table 11 – Colour depth of device icons

Icon image data		<depth> values
PNG	Grayscale: 8 bits	8
	Grayscale: 16 bits	16
	Truecolour: 24 bits (triplet of 8 bits R/G/B samples)	24
	Indexed – colour bits 24 bits (palette entry is a triplet of 8 bits R/G/B samples)	24
	Grayscale w/ alpha: 8 bits (with matching alpha channel depth)	8
	Grayscale w/ alpha: 16 bits (with matching alpha channel depth)	16
	Truecolour w/ alpha: 24 bits (triplet of 8 bit R/G/B samples, alpha channel must be 8 bits)	24
JPEG (8 bits Y/Cr/Cb samples)		24

M	R	DMS DMR DMP _r	M-DMS	n/a	IEC 62481-2 [53]

UPnP defines the way to indicate profiles for icon images.

Since <depth> value is unclear for PNG grayscale/index colored /alpha blending and JPEG, this guideline recommends the values for <depth> element required by the UPnP DA. It should be noted that the values for PNG do not help to identify color types.

7.3.30 DDC UPnP UTF-8 support

UPnP endpoints (devices and control points) must use UTF-8 encoding of all XML fragments. UPnP endpoints must be tolerant of the UTF-8 maximum of 4 bytes of Unicode character as required by XML processors.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[32] [53] [64]

Specifying UTF-8 as the encoding method for UPnP communications provides the right balance for supporting a wide variety of languages without necessarily requiring devices to support all languages.

Although UTF-8 has characters that are encoded in 6 bytes, W3C XML spec states that XML processors must accept any character in Unicode. This means XML parsers must decode up to 4 bytes of character. Specifically, see 2.2 of [64] for more information. It calls out any Unicode character, excluding the surrogate blocks, FFFE, and FFFF.

7.3.31 DDC UPnP XML comments

XML comments normally have to be skipped by XML parsers. This guideline ensures that comments do not prevent interoperation.

7.3.31.1 UPnP endpoints (devices and control points) must never source XML with comments.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

7.3.31.2 UPnP endpoints (devices and control points) may reject any XML provided with comments.

O	C	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

7.3.32 DDC UPnP boolean types

UPnP endpoints (devices and control points) must use "0" for false and "1" for true when using the UPnP boolean type.

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[53]

This simplifies control point implementations and also reduces the size of some UPnP traffic.

7.3.33 DDC CP versioning

7.3.33.1 UPnP action requests (sent by a control point) must include a DLNA-CP-version in a USER-AGENT HTTP header value.

M	A	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[27] [39] [53]

The HTTP specifications specify the format of the USER-AGENT HTTP header and header value.

It should be noted that the USER-AGENT HTTP header is not used exclusively for DLNA information.

7.3.33.2 The syntax of DLNA-CP-version is a subset of the 'product' token syntax (defined by HTTP) and is described below.

- DLNA-CP-version = "DLNADOC/" dlna-version

The dlna-version token is defined in 7.3.11.2.

Example:

- USER-AGENT: DLNADOC/1.50;
- USER-AGENT: UPnP/1.0 DLNADOC/1.50;
- USER-AGENT: CERN-LineMode/2.15 libwww/2.17b3 DLNADOC/1.50 UPnP/1.0.

M	A	DMP DMC +PR1+ +PR2+ +DN+ +PU+ +UP+	M-DMP M-DMC M- DMU, M-DMD	MIU	[27] [39]

The DLNA-CP-version token uses the syntax of a "product" token (as defined in the HTTP specifications) to identify the DLNA guidelines version.

It should be noted that space and separators cannot be used in a token, and the USER-AGENT header field uses the token syntax. For example, "My DLNA Device/1234" does not comply with the token syntax because spaces are used in the string that is supposed to follow the token syntax.

7.3.33.3 UPnP devices must be tolerant of UPnP action requests that specify a newer dlna-version in the DLNA-CP-version token.

Tolerance means that the UPnP device responds according to the version indicated by the device's <dlna:X_DLNADOC> value.

M	C	DMS DMR DMPr	M-DMS	n/a	[53]

This guideline is another clarification of the 7.3.24 in that it requires tolerance of unknown HTTP headers and values.

This guideline essentially requires a DLNA-compliant UPnP device to respond to newer control points using the DLNA-defined rules employed by the UPnP device.

7.3.34 DDC absolute and relative URI requests

7.3.34.1 The HTTP server of UPnP endpoints (devices and control points) must accept an HTTP request that specifies an absolute or relative URI in the HTTP request.

M	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

The HTTP specification indicates that this behaviour is required.

Absolute URIs are permitted in HTTP requests.

7.3.34.2 The HTTP client of UPnP endpoints (devices and control points) should specify a relative URI in the HTTP request.

S	R	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

7.3.35 DDC maximum HTTP header size

HTTP clients and servers of UPnP endpoints (devices and control points) must generate and parse HTTP messages that have a total HTTP header size that is equal to, or less than, 4096 bytes (4 KB) in all HTTP requests and responses.

The total HTTP header size is the total number of bytes from the first byte in the start-line token and the last byte of the CRLF token, as used in the generic-message token defined in 4.1 of [39], as quoted in the syntax below.

- generic-message = start-line *(message-header CRLF) CRLF [message-body]

M	L	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	[39]

This provides a reasonable assumption as to how much memory is necessary for all HTTP headers used in a single transaction related to UPnP communication.

7.3.36 DDC device capabilities

The <dlna:X_DLNAcap> is a comma-separated list of capability ID values that appear in the device description document. The syntax of the <dlna:X_DLNAcap> value, dlnacap-value, is defined as follows:

- dlnacap-value = capID *("," capID);
- capID = *("<a"-"z", "A"-"Z", "0"-"9", "_", "-" ">).

The capID token must always be a value defined by the DLNA guidelines and the length of the token must not exceed 512 bytes.

The name space for the <dlna:X_DLNAcap> must be "urn:schemas-dlna-org:device-1-0" and the namespace prefix must be "dlna:".

Example:

<dlina:X_DLNA CAP xmlns:dlina="urn:schemas-dlna-org:device-1-0">av-upload,image-upload,audio-upload</dlina:X_DLNA CAP>

M	A	DMS DMR DMPr	M-DMS	n/a	[53]

Other guidelines define the capability ID values that are permitted in the <dlina:X_DLNA CAP> element.

7.3.37 DDC DLNAQOS support

If DLNAQOS as defined in 7.2 is implemented, all UPnP device and control point traffic must be tagged with DLNAQOS_1, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), in accordance with Table 8.

M	A	HND +PR1+ +PR2+ +DN+ +PU+ +UP+	MHD	MIU	n/a

7.4 Media management

7.4.1 AV media management

7.4.1.1 General

Subclause 7.4.1 covers the guidelines for implementing media management using the UPnP AV architecture.

It is important to note that DIDL-Lite can be used in multiple contexts such as the following items.

- Some UPnP AV action request can carry DIDL-Lite metadata as input argument values (for example, CDS:CreateObject and AVT:SetAVTransportURI requests).
- Some UPnP AV action responses can carry DIDL-Lite metadata as output argument values (for example, CDS:CreateObject, AVT:GetMediaInfo, and AVT:GetPositionInfo request).
- Some UPnP AV events can carry DIDL-Lite metadata (for example, AVT.LastChange and virtual instance state variables such as AVT.CurrentTrackMetadata).

Subclause 7.4.1 AV media management guidelines is organized into multiple groups.

- Device classes and device capabilities requirements (7.4.1.2.1 to 7.4.1.2.14): This group specifies the UPnP AV components that are needed for a DLNA device class or a DLNA device capability. For example, this sub-table provides the guideline that specifies that a DMP must implement a UPnP AV MediaServer control point.
- General UPnP AV requirements (7.4.1.3.1 to 7.4.1.3.40): This group specifies general UPnP AV requirements that are used by a variety of UPnP AV devices and control points. Guidelines that dictate rules for things like DIDL-Lite metadata, protocolInfo values, and DLNA-defined parameters for the 4th field of protocolInfo values are in this sub-table.
- MediaServer requirements (7.4.1.4.1 to 7.4.1.5.4): This group provides guidelines that are specific to UPnP AV MediaServer devices. Occasionally, a related guideline that specifies behaviour for a UPnP AV MediaServer control point will also appear in this sub-table.
- MediaRenderer device requirements (7.4.1.6.1 to 7.4.1.6.22): This group provides guidelines that are specific to UPnP AV MediaRenderer devices. Occasionally, a related guideline that specifies behaviour for a UPnP AV MediaRenderer control point will also appear in this sub-table.
- Upload and optional content management (7.4.1.7.1 to 7.4.1.7.30): This group provides UPnP AV guidelines that are related to the Upload System Usage. Guidelines that specify

behaviour for CDS:CreateObject and CDS:DestroyObject transactions appear in this sub-table.

7.4.1.2 Device classes and device capabilities requirements

7.4.1.2.1 MM UPnP AV 1.0 compliance

The DLNA device classes and capabilities that implement the following device functions must fully support the mandatory portions of the UPnP AVv1 specifications: MSCP, MSD, MRCP, MRD.

M	R	DMS DMP DMR DMC +DN+ +UP+ +PU+ +PR2+	M-DMS M-DMP M- DMC M-DMD M- DMU	MIU	[50] [51] [52] [54] [55] [60]

UPnP AVv1 is the baseline architecture for discovering media sources and sinks. The applicable devices and services are described in the Device classes and device capabilities requirements section (7.4.1.2.1 to 7.4.1.2.14).

7.4.1.2.2 MM DMP/M-DMP UPnP AV MediaServer control point definition

A DMP and M-DMP must implement a UPnP AV MediaServer control point for browsing a ContentDirectory service on a DMS and M-DMS.

M	R	DMP	M-DMP	n/a	[52] [55]

This guideline indicates that a DMP device class must use a UPnP control point that controls a UPnP AV MediaServer for browsing content.

7.4.1.2.3 MM M-DMD/Download controller definition

An M-DMD and download controller must implement a UPnP AV MediaServer control point for browsing a ContentDirectory service on a DMS and M-DMS.

M	R	+DN+	M-DMD	n/a	[52] [55]

This guideline indicates that a download controller device capability must use a UPnP control point that controls a UPnP AV MediaServer for browsing content.

7.4.1.2.4 MM M-DMU/Upload controller definition

An M-DMU and upload controller must implement a UPnP AV MediaServer control point for uploading content to a DMS and M-DMS.

M	R	+UP+	M-DMU	n/a	[52] [55]

This guideline indicates that an upload controller device capability must use a UPnP control point that controls a UPnP AV MediaServer for sending content to the MediaServer.

See 7.4.1.7.3.2 and 7.4.1.7.3.3 for the functionality that must be implemented.

7.4.1.2.5 MM DMR UPnP AV MediaRenderer device definition

A DMR must implement a UPnP AV MediaRenderer device that must have a AVTransport service, RenderingControl service, and a ConnectionManager service.

M	L	DMR	n/a	n/a	[54]

DMR device must implement the baseline services for a UPnP AV MediaRenderer. This is in conjunction with the requirements for rendering endpoints described in 7.5.

7.4.1.2.6 MM DMR AVTransport rules

A DMR must support the mandatory actions and state variables for an AVTransport service.

M	R	DMR	n/a	n/a	[50]

This guideline specifies the minimum requirements for a DMR's AVTransport service.

7.4.1.2.7 MM DMR ConnectionManager rules

A DMR must support the mandatory actions and state variables for a ConnectionManager service.

M	R	DMR	n/a	n/a	[51]

This guideline specifies the minimum requirements for a DMR's ConnectionManager service.

7.4.1.2.8 MM DMR RenderingControl rules

A DMR must support the mandatory actions and state variables for a RenderingControl service.

M	R	DMR	n/a	n/a	[60]

This guideline specifies the minimum requirements for a DMR's RenderingControl service.

7.4.1.2.9 MM DMC/M-DMC UPnP AV MediaServer and AV MediaRenderer control point definition

A DMC and M-DMC must implement a UPnP AV MediaServer control point and a UPnP AV MediaRenderer control point. The MediaServer control point interacts with the ContentDirectory service for browsing content. The MediaRenderer control point interacts with the AVTransport service and the ConnectionManager service to verify that the MediaRenderer can play the content and to start and stop the playback.

M	R	DMC	M-DMC	n/a	[50] [51] [52] [54] [55]

This guideline indicates that the DMC and M-DMC device classes must use a UPnP control point that controls a UPnP AV MediaServer and a UPnP AV MediaRenderer.

7.4.1.2.10 MM push controller definition

A push controller must implement a UPnP AV MediaRenderer control point that interacts with the AVTransport service and the ConnectionManager service to verify that the MediaRenderer can play the content and to start and stop the playback.

M	R	+PU+	n/a	n/a	[50] [51] [54]

A push controller is a capability that controls a DMR and serves the content direct to the DMR. In addition to this guideline, a push controller needs to implement additional guidelines related to MediaRenderer control points and content source endpoints.

7.4.1.2.11 MM DMS/M-DMS UPnP AV MediaServer device definition

7.4.1.2.11.1 A DMS and M-DMS must implement a UPnP AV MediaServer device that must have a ContentDirectory service and a ConnectionManager service.

M	R	DMS	M-DMS	n/a	[55]

DMS and M-DMS devices must implement the minimum baseline services for a UPnP AV MediaServer.

7.4.1.2.11.2 A DMS and M-DMS may have an AVTransport service in the UPnP AV MediaServer device.

O	R	DMS	M-DMS	n/a	[55]

This guideline permits a DMS and M-DMS to implement the AVTransport service.

7.4.1.2.12 MM DMS/M-DMS ContentDirectory rules

A DMS and M-DMS must support the mandatory actions and state variables for a ContentDirectory service.

M	R	DMS	M-DMS	n/a	[52]

7.4.1.2.13 MM DMS/M-DMS ConnectionManager rules

A DMS and M-DMS must support the mandatory actions and state variables for a ConnectionManager service.

M	R	DMS	M-DMS	n/a	[51]

This guideline specifies the minimum requirements for a DMS's and M-DMS's ConnectionManager service.

7.4.1.2.14 MM MIU definition

7.4.1.2.14.1 For every virtual renderer that the MIU hosts, it must implement the media management functionality of a DMR device as defined in 7.4.1.2.5 through 7.4.1.2.8.

M	R	n/a	n/a	MIU	[54]

The MIU hosts virtual devices that control and extend the functionality of native servers and renderers in the network.

7.4.1.2.14.2 For every virtual renderer that the MIU hosts, it must implement a UPnP AV MediaRenderer control point for controlling the native DMR.

M	R	n/a	n/a	MIU	[50] [51] [54] [60]

7.4.1.2.14.3 For every virtual server that the MIU hosts, it must implement the media management functionality of either a DMS or M-DMS device as defined in 7.4.1.2.11 through 7.4.1.2.13.

M	R	n/a	n/a	MIU	[51] [52] [55]

7.4.1.2.14.4 For every virtual server that the MIU hosts, it must implement a UPnP AV MediaServer control point for controlling the native DMS or M-DMS device.

M	R	n/a	n/a	MIU	[50] [51] [52] [55]

7.4.1.3 General UPnP AV requirements

7.4.1.3.1 MM UPnP AV control point tolerance of unknown property

UPnP AV control points must be tolerant of all properties that appear in a DIDL-Lite XML fragment. Properties are as defined in Appendix B of [52].

Tolerant behaviour is defined as being able to successfully "parse and interpret" or "parse and ignore" the DIDL-Lite XML elements and attributes.

M	C	DMP DMC +DN+ +PU+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52]

This guideline ensures that a UPnP AV MediaServer control point will continue to behave properly even if a UPnP AV MediaServer device improperly implements its support of the Filter argument for CDS:Browse and CDS:Search.

7.4.1.3.2 MM DIDL-Lite restrictions

UPnP AV endpoints (devices and control points) must never employ the following in DIDL-Lite documents or fragments:

- [CDATA] payloads;
- <!--> XML comments.

DIDL-Lite documents and fragments are always assumed to have a UTF-8 encoding. The XML for DIDL-Lite documents and fragments must never contain XML comments. UPnP AV endpoints may reject any XML that is not encoded with these restrictions.

M	L	DMS DMR DMC +PU+ +UP+	M-DMS M-DMC M- DMU	n/a	[52]

The flexibility of XML can cause problems for a number of XML parsers, especially those on resource-limited platforms. These requirements balance the needs of control points with the limitations of some devices.

The assumption of using only UTF-8 encoded DIDL-Lite documents and fragments enables wide language support without requiring control points to handle multiple encoding formats.

7.4.1.3.3 MM DIDL-Lite max metadata length

7.4.1.3.3.1 Unless specified in the interoperability guidelines, element values and attribute values, appearing in DIDL-Lite documents or fragments, that are length-unlimited must not exceed 1024 bytes each, in their XML-escaped form, encoded in UTF-8.

Length-unlimited data types are the data types with an unspecified maximum length in string form. These include string, URI, bin.hex, and base64 values.

M	L	DMS DMR DMC +PU+ +UP+	M-DMS M-DMC M- DMU	n/a	[52]

This guideline puts a worst-case limit on all other metadata values found in a DIDL-Lite document or fragment. This allows for smaller limits to be specified, but that at this time this is a true maximum.

7.4.1.3.3.2 In DIDL-Lite documents or fragments, element and attribute values that are length-limited must not exceed their implied lengths.

Length-limited data types are the data types with an implied maximum length in string form. These include signed/unsigned integers, floating point numbers, boolean values, etc. The following table defines the maximum byte length for these data types that are used by the CDS and UPnP device architecture.

Table 12 – CDS and UPnP maximum byte length

Data type in CDS	Data type in UPnP device architecture	Maximum byte length
boolean	Boolean	5
unsigned integer	ui4	10
integer	i4	11
unsigned long	n/a	20
long	n/a	21
n/a	ui1	3
n/a	ui2	5
n/a	i1	4
n/a	i2	6
n/a	Int	11
n/a	r4	14
n/a	r8	22
n/a	Number	22
n/a	fixed.14.4	20
n/a	Float	110
n/a	Char	1
n/a	Date	10
n/a	dateTime	19
n/a	dateTime.tz	29
n/a	Time	8
n/a	time.tz	18
NOTE Float should be in the canonical representation.		

M	L	DMS DMP DMR DMC +DN+ +PU+ +UP+ +PR+	M-DMS M-DMP M- DMC M-DMD M- DMU	MIU	[52]

A float value is $m \times 2^e$, where m is an integer whose absolute value is less than 2^{24} , and e is an integer between -149 and 104, inclusive.

Lexical representation is as follows;

```
float-value:= mantissa [("E" | "e") exponent] | "0" | "-0" | "INF" | "-INF" | "NaN"
mantissa:= ["+" | "-"] 1*DIGIT ["." 1*DIGIT]
exponent:= ["+" | "-"] 1*DIGIT
```

The canonical representation is defined; but in the non-canonical representation, the byte length can be infinite. In the canonical representation, the maximum byte length is 110.

For example, -1E4, 1267.43233E12, 12.78e-2, 12 and INF are all legal literals for float.

See: <http://www.w3.org/TR/xmlschema-2/#float>

A boolean value can be either "0", "1", "true", or "false". The maximum length is set to 5 which is the size for the value "false". Even though guideline 7.3.34 DDC absolute and relative URI requests restricts using boolean values to "0" and "1" for DLNA device classes, it needs to be tolerant that "true" or "false" may be encountered for non-DLNA devices. Hence the reason for setting the maximum length to 5.

7.4.1.3.3.3 The dc:title metadata property should not exceed 256 bytes in the XML-escaped form encoded in UTF-8.

S	L	DMS DMP DMR DMC +DN+ +PU+ +UP+ +PR2+	M-DMS M-DMP M- DMC M-DMD M- DMU	n/a	[52]

Although the maximum length for the dc:title is 1024 bytes, many titles can fit in 256 bytes. The primary reason why title values are allowed to exceed 256 bytes is to accommodate 7.4.1.4.18.

7.4.1.3.3.4 The following metadata properties must not exceed 256 bytes each in the XML-escaped form encoded in UTF-8:

- upnp:class;
- any length-unlimited metadata property in Table 14;
- all length-unlimited DIDL-Lite schema defined attributes for <res>, except res@importUri (the length for res@importUri is governed by 7.4.1.3.10).

M	L	DMS DMP DMR DMC +DN+ +PU+ +UP+ +PR2+	M-DMS M-DMP M- DMC M-DMD M- DMU	n/a	[52]

This guideline provides devices and control points that receive metadata with some information about how much memory will be needed to represent a CDS object.

7.4.1.3.4 MM DIDL-Lite non-empty metadata values

7.4.1.3.4.1 UPnP AV endpoints that provide DIDL-Lite metadata with empty values or values composed entirely of white-spaces can cause problems for other endpoints that receive them. More importantly, a user has little idea on how to interpret such values.

7.4.1.3.4.2 UPnP AV endpoints (devices and control points) must use non-empty and non-whitespace values for metadata properties in a DIDL-Lite XML fragment. The term, metadata properties, refers specifically to elements and attributes defined by the DIDL-Lite schema. Term applies only to guideline entries of 7.4.1.3.4

NOTE This guideline specifically excludes a scenario where URI values for <res> and res@dnla:ifoFileURI specify an empty value in CDS:CreateObject requests. Until the MediaServer receives the actual content binary, the <res> URI value is permitted to be empty string. However, the MediaServer is required to specify a value for the res@dnla:importIfoFileURI in order for the control point to upload the IFO file.

M	L	DMS DMR DMC +UP+	M-DMS M-DMC M-DMU	n/a	[52]

7.4.1.3.4.3 UPnP AV endpoints (devices and control points) must not use an empty value for a corresponding DIDL-Lite attribute or element, unless explicitly required by the DIDL-Lite schema or DLNA guidelines. The empty string (represented by "" for attributes and <element></element> form for XML elements), must be used to represent an empty value.

M	L	DMS DMR DMC +UP+	M-DMS M-DMC M-DMU	n/a	[52]

7.4.1.3.4.4 UPnP AV endpoints (devices and control points) must be tolerant of DIDL-Lite attributes and elements (whether defined by DIDL-Lite schema or not) that have empty values.

M	C	DMP DMR DMC +DN+ +PU+ +UP+ +PR2+	M-DMP M-DMC M-DMU, M-DMD	MIU	[52]

7.4.1.3.5 MM DIDL-Lite boolean values

DIDL-Lite boolean values must use "0" for false and "1" for true.

M	L	DMS DMR DMC +PU+ +UP+	M-DMS M-DMC M- DMU	n/a	[52]

This simplifies control point implementations and also reduces the size of some UPnP traffic.

7.4.1.3.6 MM upnp – Class values

7.4.1.3.6.1 UPnP AV MediaServer control point must minimally treat derived classes in the same way as its ancestor class(es).

M	R	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52]

As an example, a UPnP AV MediaServer control point needs to be able to recognize a CDS object marked as an object.item.audioItem.vendorXYZ as an object.item.audioItem even though the UPnP AV MediaServer control point implementation does not understand the

meaning behind the vendorXYZ extension. It is not the intent of DLNA to require UPnP AV MediaServer control points to show all CDS objects to a user because some UPnP AV MediaServer control points may be interested in certain types of content classes.

7.4.1.3.6.2 A UPnP AV MediaServer control point must be tolerant ("parse and interpret" or "parse and ignore") of unknown upnp:class values.

M	R	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52]

Tolerance of unknown values is required, regardless of whether the control point intends to show the CDS objects to a user.

7.4.1.3.7 MM DIDL-Lite dc:date format

The syntax for the DIDL-Lite <dc:date> element value must conform to the following subset profile of ISO 8601:

- date-value = date [%x54 time [time-offset]];
- date = 4 DIGIT "-" 2 DIGIT "-" 2 DIGIT; CCYY-MMDD;
- time = 2 DIGIT ":" 2 DIGIT ":" 2 DIGIT [" " 3 DIGIT]; hh:mm:ss(.sss);
- time-offset = %x5a | ("+" | "-") 2 DIGIT ":" 2 DIGIT; Z or +hh:mm or -hh:mm.

Essentially, the following combinations are permitted:

- CCYY-MMDD;
- CCYY-MMDDThh:mm:ss;
- CCYY-MMDDThh:mm:ssZ;
- CCYY-MMDDThh:mm:ss+hh:mm;
- CCYY-MMDDThh:mm:ss-hh:mm;
- CCYY-MMDDThh:mm:ss.sss;
- CCYY-MMDDThh:mm:ss.sssZ;
- CCYY-MMDDThh:mm:ss.sss+hh:mm;
- CCYY-MMDDThh:mm:ss.sss-hh:mm.

When the offset of local time to UTC cannot be determined, the <dc:date> string must have no characters for the <time-offset> part of the date grammar.

M	R	DMS DMR DMC. +UP+ +PR2+	M-DMS M-DMC M- DMU	n/a	ISO 8601, [30]

7.4.1.3.8 MM DIDL-Lite res@duration format

The syntax of the res@duration must be compliant to the following definition:

- duration = hours ":" minutes ":" seconds;
- hours = 1*5 DIGIT; 0-99999;
- minutes = 2 DIGIT; 00-59;
- seconds = 2 DIGIT ["." 3 DIGIT]; 00-59 (,000-.999).

M	R	DMC DMS DMR +UP+ +PU+	M-DMS	MIU	[30]

7.4.1.3.9 MM DIDL-Lite desc element use

The text in the <desc> element must be a valid XML fragment with declared namespaces.

Shown below are some examples of proper <desc> usage.

- ```

<DIDL-Lite xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:upnp="urn:schemas-upnp-
org:metadata-1-0/upnp/"
 <container id="000000000000000016" searchable="1"
parentID="0000000000000000A" restricted="0" childCount="1">
 <dc:title>some title</dc:title>
 <upnp:class>
 object.item
 </upnp:class>
 <desc xmlns:example="http://some.example.org/yada/">
 <example:yada>
 some desc data
 </example:yada>
 </desc>
 </container>
</DIDL-Lite>

```
- ```

<DIDL-Lite xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"
xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:upnp="urn:schemas-upnp-
org:metadata-1-0/upnp/" xmlns:example="http://some.example.org/yada/">
  <container id="000000000000000016" searchable="1"
parentID="0000000000000000A" restricted="0" childCount="1">
    <dc:title>
      some title
    </dc:title>
    <upnp:class>object.item</upnp:class>
    <desc>
      <example:yada>
        some desc data
      </example:yada>
    </desc>
  </container>
</DIDL-Lite>

```

M	R	DMS DMR DMC +PU+ +UP+	M-DMS M-DMC M-DMU,	n/a	[52]

The XML contents of the <desc> are not validated by a DIDL-Lite validation tool, and the validity of the contained XML block is outside the scope of this standard. The industry is encouraged to utilize only valid fragments in the <desc> block.

7.4.1.3.10 MM URI rules

7.4.1.3.10.1 If a content binary conforms to a DLNA media format profile, then the URI value must be an absolute URI, with the IP address in the a.b.c.d IPv4 address format (i.e., quad-form network byte order).

M	L	DMS DMC +PR1+ +PR2+ +PU+	M-DMS M-DMC M-DMU,	n/a	[55]

		+UP+			

This guideline mandates that content URIs must use absolute URIs that use IP addresses.

7.4.1.3.10.2 URIs must be properly URI-escaped in a UTF-8 encoded form.

M	L	DMS DMC +PR1+ +PR2+ +PU+ +UP+	M-DMS M-DMC M-DMU	n/a	[25] [52]

This guideline requires UPnP AV endpoints to provide URI values that are URI-escaped in a UTF-8 encoded form. This guideline also allows a UPnP AV MediaServer control point to assume that URIs obtained from a UPnP AV MediaServer do not need any escaping.

7.4.1.3.10.3 HTTP URI escaping must be performed according to the URI specification ([25]) as required in 3.2.1 of the HTTP/1.1 specification [39].

M	L	DMS DMC +PR1+ +PR2+ +PU+ +UP+	M-DMS M-DMC M-DMU	n/a	[25] [39]

7.4.1.3.10.4 URI values that appear in DIDL-Lite documents or fragments must not exceed 1024 bytes, in the URI-escaped UTF-8 encoded form.

URI values must not have preceding or trailing white space characters.

M	L	DMS DMC +PR1+ +PR2+ +PU+ +UP+	M-DMS M-DMC M-DMU	n/a	[25] [52] [64]

URI values are theoretically infinite in length. This guideline puts a reasonable limit on the length of advertised content URI values.

7.4.1.3.10.5 If a content binary does not conform to a DLNA media format profile, then the URI value may be a URI, with a domain name.

O	L	DMS DMC +PR1+ +PR2+ +PU+ +UP+	M-DMS M-DMC M-DMU	n/a	[55]

This guideline permits the use of content URIs that use domain names when content is not marked as conformant to a DLNA media format profile. Content sourced from the Internet is considered outside the scope for DLNA. However, a ContentDirectory service still has a way of advertising Internet content in a DLNA manner using IPv4 addresses.

7.4.1.3.11 MM DIDL-Lite recommended metadata properties

7.4.1.3.11.1 Content that conforms to the DLNA "Image" media class must use object.item.imageItem or a derived class for the upnp:class value.

Content that conforms to the DLNA "Audio" media class must use object.item.audioItem or a derived class for the upnp:class value.

Content that conforms to the DLNA "AV" media class must use object.item.videoItem or a derived class for the upnp:class value. (A CDS object with the object.item.videoItem or derived class value may have image-based thumbnails as described by 7.1.5 JPEG TN Format Profile and 7.2.1 PNG TN Format Profile in IEC 62481-2.)

M	A	DMS +UP+	M-DMS M-DMU,	n/a	IEC 62481-2, [52]

7.4.1.3.11.2 Metadata properties defined in Appendix B of the ContentDirectory service specification [52] must use the following namespace prefixes.

Table 13 – Namespace prefixes

Namespace	Prefix
DIDL-Lite	n/a
Dublin core	dc:
UPnP	upnp:

Reference [52] contains Appendix B as noted.

M	L	DMS +UP+	M-DMS M-DMU,	n/a	[52]

n/a means that the prefix is not used.

7.4.1.3.11.3 A UPnP AV MediaServer device should provide non-empty and non-whitespace values for metadata properties as shown in the table below for the purpose of content selection.

Table 14 – Recommended metadata properties

upnp:class value	Property names
object.item.audioItem	dc:creator, upnp:album, upnp:genre, res@duration, res@size
object.item.imageItem	dc:date, res@resolution, res@size
object.item.videoItem	dc:date, upnp:genre, res@duration, res@size
object.container.album.musicAlbum	dc:creator, upnp:genre, @childCount
object.item.videoItem.videoBroadcast object.item.audioItem.audioBroadcast	upnp:genre, upnp:channelName, upnp:channelNr (Applicability of upnp:channelNr depends on region)

This guideline also applies to classes derived from those listed in the table.

The dc:creator property is understood to contain a value representing the artist name or some equivalent content creator. Whenever possible, the original artist name or content creator should be used for dc:creator. In some cases, such as an audio playlist, a user is the original content creator of the media collection. While it may be appropriate to specify the user's name as the dc:creator for a playlist, it is not recommended to specify the user's name as the dc:creator value for individual audio items in the playlist.

S	A	DMS	M-DMS	n/a	[52]

This guideline recommends that some additional metadata properties be used for different media classes. Although not required, providing the user with an information-rich user experience is desirable.

When an upload controller or M-DMU creates an object on the MediaServer it should provide these values as part of the DIDL-Lite in the CDS:CreateObject request; see guidelines 7.4.1.7.23.9 and 7.4.1.7.24.4 for this requirement.

7.4.1.3.11.4 If a UPnP AV MediaServer control point lists CDS objects with a particular base-class upnp:class value to users, then it must also list CDS objects that have a upnp:class value that is derived from the supported base-class.

For example, a MediaServer control point displays content listings for images with a upnp:class value object.item.image. Item needs also to display CDS objects with a upnp:class value of object.item.imageItem.xyz.

M	C	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M-DMD M-DMU	n/a	[52]

Control points must not match exclusively on base-class values because guideline 7.3.26.1 allows MediaServers to use derived-class values. Derived-class values allow a DMS to provide more information about a CDS object. Control points can use this additional information in a variety of ways, but using the upnp:class value to prevent users accessing valid content is not acceptable behaviour.

7.4.1.3.12 MM protocolInfo context

7.4.1.3.12.1 For all guidelines related to protocolInfo values, the following interpretation must be applied when applying a context for the protocolInfo value. Additions, exceptions, and other clarifications to these baseline interpretation rules must be indicated in other guidelines on a per parameter basis.

M	C	DMP DMS DMR DMC +DN+ +PU+ +UP+ +PR2+	M-DMP M-DMD M-DMS M-DMU	MIU	[50] [51] [52]

7.4.1.3.12.2 The phrase “match by protocolInfo format” must mean the following.

If given two protocolInfo values, the protocolInfo values “match by protocolInfo format” if they both have the same values for the following:

- first field of protocolInfo, except as noted below;
- in order to match, equality for the first field is required for all scenarios, except for those involving upload AnyContainer or other optional content management (OCM) operations;
- DLNA.ORG_PN parameter in the fourth field of protocolInfo.

M	C	DMP DMS DMR DMC +DN+ +PU+ +UP+ +PR2+	M-DMP M-DMD M-DMS M-DMU	MIU	[50] [51] [52]

This phrase is used throughout the guidelines related to protocolInfo values. The phrase applies to any protocolInfo value, regardless from where the values were obtained. ProtocolInfo values can be found on DMS and DMR devices, typically associated with res@protocolInfo metadata or UPnP AV connection information.

This phrase applies only to protocolInfo values that identify a DLNA media format profile.

7.4.1.3.12.3 If a UPnP AV MediaServer or UPnP AV MediaRenderer exposes a `res@protocolInfo`, then the context of the `protocolInfo` must be to describe the content and/or transport layer features for the indicated `<res>` URI value.

M	C	DMS DMR	M-DMS	n/a	[50] [51] [52]

DMS devices report `res@protocolInfo` values when responding to `ContentDirectoryService` requests, including those related to the upload system usage or optional content management operations.

A DMR can create DIDL-Lite metadata to describe what it is currently rendering. See 7.4.1.3.12.11 for more information.

7.4.1.3.12.4 If a UPnP AV MediaServer exposes a `res@protocolInfo` with a `<res>` element that omits the URI value, then the context of the `protocolInfo` must be to describe the content that MediaServer expects to receive in a content transfer process.

M	C	DMS	M-DMS	n/a	[51] [52]

When used with a `<res>` element that omits a URI value, the `DLNA.ORG_PN` parameter identifies the DLNA media format profile that the DMS expects to receive for that particular `<res>` element. In some cases, DMS implementations will specify various parameters and flags in the 4th field that will become applicable after the content is uploaded and the DMS can serve the content.

7.4.1.3.12.5 If a UPnP AV MediaServer exposes a `res@protocolInfo` with a `<res>` element that omits the URI value, then the `protocolInfo` 4th field may contain parameters and flags intended for use when the `<res>` URI value is finally created by the MediaServer.

O	C	DMS	M-DMS	n/a	[51] [52]

7.4.1.3.12.6 If a UPnP AV MediaServer lists a `protocolInfo` in the `CMS.SourceProtocolInfo` state variable, then the context of the `protocolInfo` must be to describe the MediaServer's ability to support the described feature or content for one or more of the following:

- serving content to other endpoints on the network;
- upload AnyContainer and possibly other optional content management (OCM) operations that involve content that “match by `protocolInfo` format”.

(i.e. The parameters and flags represent a union of the MediaServer's features/capabilities for the specified DLNA media format profile.)

This guideline works in conjunction with 7.4.1.3.13.

M	C	DMS	M-DMS	n/a	[51] [52]

The phrase “MediaServer's ability to support the described feature or content” means that the DMS can serve content binaries (or receive content binaries in a content transfer process) that are characterized by the `protocolInfo`. More specifically, for each `protocolInfo` listed in `SourceProtocolInfo` the DMS exposes zero or more content binaries that “match by `protocolInfo` format”. (The zero case is not to be used as a loop-hole for listing media format profiles that the DMS will never advertise.)

For example, if a MediaServer lists "http-get:*:audio/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;DLNA.ORG_OP=01;DLNA.ORG_FLAGS=AD5000000000000000000000000000" then the MediaServer is capable of serving MPEG_PS_NTSC content with the following characteristics.

- Some of that content may support the range HTTP header under the "full random access data availability" model due to the DLNA.ORG_OP value.
- Some of that content may support the range HTTP header under the "limited random access data availability" model because the lop-bytes flag is true.
- Some of that content may have the sp-flag set to true.
- Some of the content may have a beginning point that increases with time (for example, live content) because the s₀-increasing flag is true.
- Some of the content may have an ending point that increases with time (for example, infinitely long live content or live content that is currently being saved to disk and will eventually reach a finite length) because the s_N-increasing flag is true.
- Some of the content may support streaming and background transfer modes because the tm-s and tm-b flags are true.

It should be noted that the pn-param (DLNA.ORG_PN) is the only required parameter for DLNA media format profiles. All other parameters and flags are optional for the syntax.

7.4.1.3.12.7 If a UPnP AV MediaRenderer lists a protocolInfo in the CMS.SinkProtocolInfo state variable, then the context of the protocolInfo must be to describe the MediaRenderer's ability to support that the indicated feature or content (i.e., the parameters and flags represent a union of the MediaRenderer's features/capabilities for the specified DLNA media format profile).

This guideline works in conjunction with 7.4.1.3.13.

M	C	DMR	n/a	n/a	[50] [51]

The phrase "MediaRenderer's ability to support that the indicated feature or content" means that the DMR can use the specified feature when rendering content that is a "match by protocolInfo format". It is not a guarantee that the DMR will use that feature when rendering such content. Each DMR implementation determines when and how features are used to deliver a playback experience.

For example, if a MediaRenderer lists "http-get:*:audio/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;DLNA.ORG_OP=01;DLNA.ORG_FLAGS=BD1000000000000000000000000000" then the MediaRenderer is capable of rendering MPEG_PS_NTSC content for a variety of scenarios.

- It can render MPEG_PS_NTSC content regardless of whether the sp-flag, s₀-increasing, or s_N-increasing flags are set to true or false for that content.
- If the content source makes the range HTTP header available, the content receiver is able to use the range HTTP header for both "full random access data availability" and "limited random access data availability" models because of the op-param and the lop-bytes flag.
- If given a PlayContainer URI, the DMR will also play MPEG_PS_NTSC content found during the traversal of the DMS because the playcontainer-param is set to true.
- The DMR uses streaming transfers for the MPEG2_PS_NTSC content, as required for immediate rendering.

It should be noted that the pn-param (DLNA.ORG_PN) is the only required parameter for DLNA media format profiles. All other parameters and flags are optional for the syntax.

7.4.1.3.12.8 If a UPnP AV MediaServer or UPnP AV MediaRenderer uses a `protocollInfo` value in the `protocollInfo` output argument of a `CMS:GetCurrentConnectionInfo` response, then the context of the `protocollInfo` must be to describe the content and the transport layer capabilities of the UPnP AV connection.

M	C	DMS DMR	M-DMS	n/a	[50] [51] [52]

In this context, the `protocollInfo` value describes the content and/or transport layer capabilities of the UPnP AV connection from the server-side.

For example, if a UPnP AV connection indicates "`http-get:*:audio/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;DLNA.ORG_OP=01;DLNA.ORG_FLAGS=05100000000000000000000000000000`" then the transport layer associated with the UPnP AV connection is configured to work in the following ways.

- The `MPEG_PS_NTSC` content has the `sp-flag` set to false and the `tm-s` flag is set to true to indicate that transport layer is using a streaming transfer where the content source is not the clock source. This means the content source will ensure that it transmits at a throughput necessary for rendering but will also preserve the entire content bitstream even for lower transmission throughputs when network conditions are not able to support the bandwidth needed for the streaming transfer.
- The range HTTP header is available only under the "full random access data availability" model.
- The content is growing with a fixed beginning because the `s0-increasing` flag is false and the `sN-increasing` flag is true.

The phrase "describe the content and the transport layer capabilities of the UPnP AV connection" does not imply that actual data is actually being transported on the UPnP AV connection at the current moment because the content receiver endpoint may be in a playback state (such as stop or pause) that does not involve the ongoing transmission of content data.

7.4.1.3.12.9 If a UPnP AV MediaServer control point creates a `res@protocollInfo` value for use with a `CDS:CreateObject` request that qualifies as one of the listed operations, then the context of the `protocollInfo` must be to describe the content that is going to be uploaded to the MediaServer through a content transfer process:

- upload AnyContainer;
- OCM: upload content.

Furthermore, the default behaviour of the control point must be to exclude 4th field parameters (except the `DLNA.ORG_PN` parameter, described in 7.4.1.3.18.1; The syntax definition of `pn-param` must be as follows:) or specify the parameter with a false value (such as in the case of the `lop-npt` flag, described in 7.4.1.3.28).

M	C	+UP+	M-DMU	MIU	[52]

Unless specified otherwise, 4th field parameters do not apply to what the upload controller can do. Rather, the DMS will provide the appropriate 4th field parameters during the course of an upload system usage or an optional content management operation. See 7.4.1.3.12.4 and 7.4.1.3.12.5 for more information.

7.4.1.3.12.10 If a UPnP AV MediaRenderer control point creates a `protocollInfo` or `res@protocollInfo` value, then the context of the `protocollInfo` must be to describe the content source's capabilities for the associated UPnP AV connection or `<res>`.

NOTE "MediaRenderer control point creates a `protocollInfo` or `res@protocollInfo`" specifically refers to the following scenarios. This phrase is used in other guidelines related to `protocollInfo`.

- `res@protocollInfo` values that appear in CDS or DIDL-Lite metadata used as input arguments for `AVT:SetAVTransportURI`, or other UPnP AV actions.

M	C	DMC +PU+	M-DMC	MIU	[50] [51] [52]

In the case of `AVT:SetAVTransportURI`, the `res@protocollInfo` can appear in the DIDL-Lite metadata of the `CurrentURIMetaData` input argument.

7.4.1.3.12.11 A UPnP AV MediaRenderer may use `res@protocollInfo` values in the following ways:

- `res@protocollInfo` values may appear in any AVTransport virtual instance state variable, including but not limited to: `AVT.NextAVTransportURIMetaData`, `AVT.CurrentTrackMetadata`, and `AVT.AVTransportURIMetaData`;
- `res@protocollInfo` values may appear in output arguments of any AVTransport actions, including but not limited to
 - `CurrentURIMetaData` of `AVT:GetMediaInfo`,
 - `NextURIMetaData` of `AVT:GetMediaInfo`, or
 - `TrackMetadata` of `AVT:GetPositionInfo`.

O	C	DMR	n/a	n/a	[50]

This guideline clarifies where a DMR can use `res@protocollInfo` values.

7.4.1.3.13 MM consistency of `protocollInfo` state variables and output arguments

If the listed values for the `CMS.SinkProtocollInfo` or `CMS.SourceProtocollInfo` state variables change, then the `ConnectionManager` service must report the change through the GENA event mechanism, as defined by [53].

`CMS:GetProtocollInfo` must return the same `protocollInfo` list for `SinkProtocollInfo` output argument as the current list indicated by the `CMS.SinkProtocollInfo` state variable.

`CMS:GetProtocollInfo` must return the same `protocollInfo` list for `SourceProtocollInfo` output argument as the current list indicated by the `CMS.SourceProtocollInfo` state variable.

M	R	DMS DMR	M-DMS	n/a	[53] [51]

Many 4th field DLNA guidelines refer to `CMS:GetProtocollInfo` arguments, without referring specifically to the associated `ConnectionManager` service state variables. This guideline repeats the `ConnectionManager` service's requirement that `CMS:GetProtocollInfo` accurately represent the data in the associated state variables, which in turn allows the DLNA syntax rules to apply to CMS state variables.

7.4.1.3.14 MM `CMS:GetProtocollInfo` rules

7.4.1.3.14.1 The `ConnectionManager` service must list the union set of `protocollInfo` values supported by the device.

If a `ConnectionManager` reports support for a DLNA media format profile, it must explicitly list the appropriate `protocollInfo` values with the `DLNA.ORG_PN` parameter in the 4th field.

M	R	DMS DMR	M-DMS	n/a	[51]

This guideline makes it easier for control points to find servers that provide content for a given profile. In the case of MediaRenderers, the protocolInfo values indicate what can be played on the device.

Use of the 'http-get:*:*:*' protocolInfo does not sufficiently communicate the supported media format profiles. This guideline also requires implementations to explicitly list the individual protocolInfo values for supported DLNA media format profiles.

7.4.1.3.14.2 The sets of protocolInfo values returned in CMS:GetProtocolInfo must list the protocolInfo values that use the http-get media transport and DLNA media format profiles first.

M	L	DMS DMR	M-DMS	n/a	[51]

Although the number of returned protocolInfo values may be large, control points may use this guideline to capture the subset of relevant DLNA protocolInfo values.

7.4.1.3.14.3 The 4th field of the protocolInfo values (obtained from the ConnectionManager service) that have the DLNA.ORG_PN parameter in the 4th field must also provide the following parameters, if the corresponding operations are supported for the indicated media format profile:

- DLNA.ORG_PS;
- DLNA.ORG_OP;
- DLNA.ORG_MAXSP;
- DLNA.ORG_FLAGS.

M	A	DMR	n/a	n/a	[51]

This guideline works in conjunction with 7.4.1.3.12.7, which says that the parameters for the protocolInfo represent a union of the DMS's capabilities.

The DLNA.ORG_PS parameter means the MediaRenderer supports requests with the PlaySpeed.dlna.org header in the case of HTTP and SCALE in the case of RTP media transport.

The DLNA.ORG_OP parameter means the MediaRenderer supports requests with the range and/or TimeSeekRange.dlna.org HTTP header.

The DLNA.ORG_MAXSP parameter means the MediaRenderer supports requests with the SPEED header for RTP media transport.

7.4.1.3.15 MM DIDL-Lite protocolInfo values

7.4.1.3.15.1 If the <res> value contains an HTTP URL, then the first field of the res@protocolInfo value must be as shown below.

- http-get

M	R	DMS DMR +PU+	M-DMS	n/a	[52]

Guidelines in 7.4.1.3.15 provide requirements on the protocolInfo values in DIDL-Lite documents.

For requirements around protocolInfo values used in the upload process, see 7.4.1.7.19.1.

Reiterates the protocol string value for DLNA content transported across HTTP. Note that the converse is not always true. Some scenarios involving upload operations can result with a <res> element that has no value (i.e. empty URI) but the res@protocolInfo value has "http-get" in the first field.

7.4.1.3.15.2 If the <res> value contains an RTP URL, then the first field of the res@protocolInfo value must be as shown below.

- **rtsp-rtp-udp**

M	R	DMS DMR +PU+	M-DMS	n/a	[52]

7.4.1.3.15.3 The third field of a protocolInfo value must use the MIME types specified in the compendium of media format profiles, Clause 5 of IEC 62481-2 for DLNA normative media format profiles when the protocol is "http-get" or "rtsp-rtp-udp".

M	F	DMS DMR DMC +PU+	M-DMS M-DMC	n/a	IEC 62481-2, [52]

This guideline is consistent with the UPnP AV specifications for the case of "http-get", but this guideline also conflicts in the case of "rtsp-rtp-udp".

The UPnP AV specification indicates that the 4th field for RTP URLs is the payload type, but such a model fails to take into account that streaming of AV content with RTP generally requires multiple RTP payload types. For this reason, the DLNA guidelines require that the 3rd field is populated with the DLNA-specified mime-type of the media format profile indicated in the pn-param.

7.4.1.3.15.4 UPnP AV endpoints (devices and control points) must use the DLNA.ORG_PN parameter (7.4.1.3.17) in the 4th field of protocolInfo for the following operations:

- to identify content conforming to a DLNA media format profile; or
- to specify that a UPnP device is capable of receiving, distributing, or rendering content conforming to a DLNA media format profile.

M	L	DMS DMP DMR DMC +DN+ +PU+ +UP+ +PR2+	M-DMP M-DMS M-DMC M-DMU	MIU	[52]

MIME-types in the 3rd field of res@protocolInfo alone are not sufficient to identify DLNA content.

MIME-types are not to be used to identify DLNA media format profiles because MIME-types will be updated by IANA.

7.4.1.3.15.5 If a protocolInfo value has the first value of "http-get" and the 4th field includes the DLNA.ORG_PN parameter (7.4.1.3.17), then the binary data conformant to a DLNA media format profile identified by the DLNA.ORG_PN parameter must be transmitted via the HTTP media transport, without transport layer encryption.

M	L	DMS DMP DMR DMC +DN+ +PU+ +UP+ +PR2+	M-DMP M-DMS M-DMC M-DMU M-DMC	MIU	[51]

This means that the binary data via the HTTP media transport is not encrypted unless the guidelines of the DLNA media format profile explicitly states it.

EXAMPLE 1:

The following are example scenarios of creating 4th field values and the DLNA device classes capabilities that apply to them.

- [C1] DMS, M-DMS, MIU: These endpoints populate res@protocolInfo in metadata responses and protocolInfo used in CMS:GetProtocolInfo.
- [C2] DMS, M-DMS, +PU+ +PR1+, MIU: These endpoints populate values for contentFeatures.dlna.org at the transport layer, either by creating a brand new 4th field value or modifying from existing 4th field value.
- [C3] DMC, M-DMC, +PU+, MIU: The control points of these endpoints can create a 4th field value for metadata given to a DMR, either by creating a brand new 4th field value or modifying an existing 4th field value.
- [C4] DMR, MIU: When reporting metadata of content that is currently playing, these endpoints can create a new 4th field value by modifying an existing 4th field value acquired from a DMS, DMC, or +PU+.
- [C5] M-DMU, +UP+, MIU: These endpoints create res@protocolInfo values in CDS:CreateObject requests and also create the contentFeatures.dlna.org value during the content transfer process.

EXAMPLE 2:

The following are example scenarios of parsing 4th field values, and the DLNA device classes capabilities that apply to them.

- [P1] DMP, M-DMP, M-DMD, DMR, DMC, M-DMC, M-DMU, +UP+, +PR2+, +DN+, MIU: Control points of these endpoints parse 4th field values in metadata responses obtained from a DMS or M-DMS.
- [P2] DMP, M-DMP, M-DMD, DMR, DMPr, +DN+, MIU: These endpoints parse 4th field values from contentFeatures.dlna.org, at the transport layer.
- [P3] DMR, MIU: These endpoints can receive 4th field values in DIDL-Lite metadata, provided by a control point.
- [P4] DMC, M-DMC, +PU+, MIU: The control points of these endpoints can receive 4th field values in DIDL-Lite metadata that is exposed by a DMR.
- [P5] DMS, M-DMS, MIU: These endpoints parse res@protocolInfo values in CDS:CreateObject requests and also parse the contentFeatures.dlna.org value during the content transfer process.

7.4.1.3.16 MM protocolInfo values – 4th field

7.4.1.3.16.1 If a protocolInfo value has "http-get" as the first field value and the 4th field includes the pn-param token, then the following syntax must be used for the 4th field.

- 4th_field = pn-param [op-param] [ps-param] [ci-param] [flags-param] [*(other-param)]

NOTE In all guidelines that relate to the syntax of the 4th field, the relative order of pn-param, op-param, ps-param, ci-param, flags-param, and *(other-param) used in 4th_field is mandatory. For example, pn-param cannot appear after op-param, ps-param, ci-param, flags-param, or *(other-params).

The syntax and definition of pn-param, op-param, ps-param, ci-param, flags-param, and *(other-param) are defined in the guidelines below.

M	A	DMP DMS DMC DMR DMPr +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This guideline defines the syntax of the fourth field for a res@protocolInfo value that indicates content that is transported across HTTP.

Note that this syntax prohibits the use of the "*" value for content that conforms to a DLNA media format profile. Content that does not conform to a DLNA media format profile can use the "*" value in the 4th field.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4].

Examples [C5] and [P5] do not apply to this guideline because 7.4.1.7.19 covers those cases.

7.4.1.3.16.2 If a protocolInfo value has "rtsp-rtp-udp" as the first field value and the 4th field includes the pn-param token, then the following syntax must be used for the fourth field.

- 4th_field-rtsp = pn-param [op-param] [ps-param] [ci-param] [flags-param] [maxsp-param] [* (other-param)]

M	A	DMS DMP DMR DMC +DN+ +PU+ +UP+ +PR2+	MHD	MIU	[50] [51] [52]

The 4th field syntax for rtsp-rtp-udp URIs is the same as for http-get URIs.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4].

NOTE DMPr and +PR1+ are not listed for this guideline because RTP Media Transport is never used to transport images. However, M-DMD, +DN+, and +PR2+ are listed because the associated control points are required to tolerate <res> elements that use the RTP Media Transport when browsing a MediaServer.

7.4.1.3.16.3 If a protocolInfo value has the first field value that is neither equal to "http-get" nor "rtsp-rtp-udp", then the fourth field may have a syntax that differs from 7.4.1.3.16.1 and 7.4.1.3.16.2.

O	A	DMP DMS DMC DMR DMPr +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This guideline permits a different syntax for the fourth field of a protocolInfo value when the content is not transported over a DLNA-specified transport.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4].

The CDS:CreateObject syntax has restrictions on the 4th field syntax, as described in 7.4.1.7.19.

Examples of protocolInfo values that include the 4th field are shown below:

- http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;
- http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_PAL;DLNA.ORG_OP=10;
- http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;DLNA.ORG_OP=10
DLNA.ORG_PS=-1,30,-30;
- http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_PAL;DLNA.ORG_PS=-1,30,-30;
- http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_PAL
DLNA.ORG_PS=-1,30,-30;VENDORXYZ.COM_FOO=bar;
- http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_PAL;DLNA.ORG_CI=1
VENDORXYZ.COM_FOO=bar;

- rtsp-rtp-udp*:video/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;DLNA.ORG_OP=10
DLNA.ORG_PS=-1,2/3,4;DLNA.ORG_FLAGS=03100000000000000000000000000000
DLNA.ORG_MAXSP=9.75.

7.4.1.3.17 MM pn-param (DLNA.ORG_PN Parameter)

The syntax definition of pn-param must be as follows:

- pn-param = "DLNA.ORG_PN=" pn-value;
- pn-value = *("<"a"-"z", "A"-"Z", "0"-"9", "_")>.

The pn-value must identify the DLNA media format profile ID that is applicable for the context of the protocolInfo. (See 7.4.1.3.12 for determining an appropriate context for the protocolInfo.)

The pn-param is reserved for use with contexts where content conforms to a DLNA media format profile. Use of pn-param for content not conformant with a DLNA media format profile is expressly prohibited.

M	A	DMP DMS DMC DMR DMPr +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This guideline defines the syntax of the DLNA.ORG_PN parameter. This parameter is used to identify DLNA content. This parameter cannot be used for non-DLNA content.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [C5], [P1], [P2], [P3], [P4], and [P5].

7.4.1.3.18 MM op-param (operations parameter – common guidelines)

7.4.1.3.18.1 The syntax definition of op-param must be as follows:

- op-param = [op-param-delim] "DLNA.ORG_OP=" op-value;
- op-param-delim = ";";
- op-value = a-val b-val;
- a-val = boolean;
- b-val = boolean;
- boolean = "1" | "0".

The op-value is a string composed of two characters: a-val and b-val. The meaning of these values is described in these guidelines, depending on whether the context is for the HTTP media transport or RTP media transport.

- 7.4.1.3.19 MM op-param (operations parameter for HTTP)
- 7.4.1.3.20 MM op-param (operations parameter for RTP)

If the first field of protocolInfo is neither "http-get" nor "rtsp-rtp-udp" then the fourth field must omit the op-param.

M	A	DMP DMS DMC DMR DMPr +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This guideline defines the DLNA.ORG_OP parameter.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4].

7.4.1.3.18.2 If the op-param is present and if either a-val or b-val is "1", then the "full random access data availability" model must be the data access model that applies in the context of the protocolInfo value.

Specifically, this means that the transport operation (that is indicated by the a-val or b-val) must be supported for the entire content binary, as defined in 7.5.4.2.15 MT "full random access data availability" model.

If the flags-param token is included in the 4th field, then the s₀-increasing, lop-npt, and lop-bytes bits of the primary-flags token must be set to false.

M	C	DMS +PU+ +PR1+	M-DMS	MIU	[50] [51] [52]

The DLNA.ORG_OP parameter indicates support for transport layer headers responsible that facilitate random access operations on content binaries, under the "full random access data availability" model.

For more information on the "full random access data availability" model, see the following guidelines.

7.5.4.2.14 MT normative random access data availability models

7.5.4.2.15 MT "full random access data availability" model

The "full random access data availability" model is mutually exclusive with the "limited random access data availability" model, which is why lop-npt/lop-bytes cannot be used with the op-param. For more information, see the following guidelines.

7.4.1.3.28 MM lop-npt and lop-bytes (limited operations flags) – Common

7.5.4.2.16 MT "limited random access data availability" model

The following examples apply to this guideline: [C1] and [C2].

7.4.1.3.18.3 In conjunction with the rules defined in 7.4.1.3.18.1 and 7.4.1.3.18.2, the fourth field of a res@protocolInfo may use the op-param for non-DLNA media format profiles.

O	A	DMS +PU+ +PR1+	M-DMS	MIU	[50] [51] [52]

This guideline permits the use of DLNA.ORG_OP for both DLNA and non-DLNA content, provided the DLNA-defined syntax and semantics are used.

The following examples apply to this guideline: [C1] and [C2].

7.4.1.3.18.4 If the b-val token is true, then the content source should provide content length information in the res@size value of the CDS object.

S	L	DMS +PU+	M-DMS	MIU	[52]

This guideline helps resolve dependency issues that content receivers have on knowing the content length, and where content length is not provided in a chunked response. It is also recommended that content sources provide a res@size value even in scenarios where byte range is not supported.

The following examples apply to this guideline: [C1] and [C2]. It should be noted that this guideline does not apply to +PR1+ because it does not advertise content with res@size.

7.4.1.3.18.5 If the a-val token is true, then the content source should provide duration information in the res@duration value of the CDS object.

S	L	DMS +PU+	M-DMS	MIU	[52]

This guideline resolves dependency issues that content receivers have on knowing the content duration. If content sources support TimeSeekRange.dlna.org, duration information should be known, so content sources should provide it.

The following examples apply to this guideline: [C1] and [C2]. It should be noted that this guideline does not apply to +PR1+ because it does not advertise content with res@size.

7.4.1.3.18.6 If the pause media operation is supported, rendering endpoints must support the pause release media operation even in the absence of content length, res@size, or res@duration information.

M	L	DMR DMP	M-DMP M-DMD	MIU	[52]

Rendering endpoints can issue a range request using the "beginrange-" notation so as to avoid specifying an invalid range. Rendering endpoints can issue a TimeSeekRange.dlna.org request using the "begintime-" notation so as to avoid specifying an invalid range.

The variants of examples [P1] and [P2] that involve rendering of audio or AV content apply to this guideline.

7.4.1.3.19 MM op-param (operations parameter for HTTP)

7.4.1.3.19.1 If the 4th field is associated with the "http-get" transport protocol, then the a-val and b-val tokens (of the op-param token) mean the following.

- a-val indicates support of the TimeSeekRange.dlna.org HTTP header (see 7.5.5.2.24 MT HTTP time-based seek (server)) for the context of the protocolInfo under the "full random access data availability" model.
- b-val indicates support of the range HTTP header (see 7.5.5.2.22 MT HTTP header: range (server)) for the context of the protocolInfo under the "full random access data availability" model.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	[39] [51] [52]

This guideline defines the op-param token's a-val and b-val tokens when HTTP is the transport protocol.

When used with a context involving HTTP, the DLNA.ORG_OP parameter identifies if the server supports the TimeSeekRange.dlna.org or range HTTP headers for the associated content binary under the "full random access data availability" model.

For more information on the "full random access data availability" model for HTTP and these HTTP headers, see the following guidelines.

7.5.4.2.14 MT normative random access data availability models

7.5.4.2.15 MT "full random access data availability" model

7.5.5.2.18 MT HTTP common random access data availability requirements

7.5.5.2.19 MT HTTP data range of "full random access data availability"

The following examples apply to this guideline: [C1] and [C2]. Note that the TimeSeekRange.dlna.org header never applies to scenarios involving XHTML print documents and images.

7.4.1.3.19.2 If the associated HTTP server endpoint returns HTTP error code 406 (not acceptable) because an HTTP specifies one of the above HTTP headers, then the op-param must indicate that the HTTP header is not supported by using a value of "0" for the appropriate a-val or b-val.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	[39] [51] [52]

HTTP server endpoints use the 406 (not acceptable) status code to indicate that an HTTP request can never be satisfied with the specified HTTP headers.

The following examples apply to this guideline: [C1] and [C2].

7.4.1.3.19.3 If the associated HTTP server endpoint always returns 406 (not acceptable) in response to requests that use either HTTP header for the context of the protocolInfo, then the 4th field must omit the op-param.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	[39] [51] [52]

Including an op-param with a value of "00" is prohibited. In cases where neither the TimeSeekRange.dlna.org nor the range header are supported, the op-param is omitted.

The following examples apply to this guideline: [C1] and [C2].

7.4.1.3.19.4 If the associated HTTP server endpoint is capable of responding with a target response that appropriately corresponds to the data range indicated in the HTTP request's TimeSeekRange.dlna.org header value, then the a-val token must be true (i.e. the associated HTTP server endpoint does not return error code 406 (not acceptable), unless other HTTP headers are the cause of the error.)

Lastly, the content source must be able to support TimeSeekRange.dlna.org on the entire content binary, as required by 7.4.1.3.18.2.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	[39] [51] [52]

An HTTP Server that can respond with content data occupying a particular npt time range needs to advertise this capability in the 4th field.

The following examples apply to this guideline: [C1] and [C2].

Note that "appropriately corresponds" is clarified by 7.5.5.2.24.6 (MT HTTP time-based seek (server)), which permits recommends returning data from a decoder-friendly point.

7.4.1.3.19.5 If the associated HTTP server endpoint is capable of responding with a target response that corresponds exactly to the data range indicated in the HTTP request's range header value, then the b-val token must be true (i.e., the associated HTTP server endpoint does not return error code 406 (not acceptable), unless other HTTP headers are the cause of the error.)

Lastly, the content source must support the range on the entire content binary, as required by 7.4.1.3.18.2.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	[39] [51] [52]

An HTTP server that can respond with content data occupying a particular byte range needs to advertise this capability in the 4th field.

The following examples apply to this guideline: [C1] and [C2].

7.4.1.3.19.6 If the associated HTTP server endpoint supports the `realTimeInfo.dlna.org` HTTP header with a finite `max-lag-time` value (as described in 7.5.5.3.18 MT HTTP header: `realTimeInfo.dlna.org`), then the `op-param` must be omitted.

M	A	DMS +PU+	M-DMS	MIU	[39] [51] [52]

The following examples apply to this guideline: [C1] and [C2]. It should be noted that this guideline never applies to printing related scenarios, which is why `+PR1+` is excluded from this guideline.

7.4.1.3.20 MM op-param (operations parameter for RTP)

7.4.1.3.20.1 If the 4th field is associated with the "rtsp-rtp-udp" transport protocol, then the `a-val` and `b-val` tokens (of the `op-param` token) mean the following.

- `a-val` indicates support of the range header (see 7.5.8.2.39 MT RTP receiving endpoint range header and 0 MT RTP serving endpoint range header) for the context of the `protocolInfo` under the "full random access data availability" model.
- `b-val`, in scenarios involving the RTP Media Transport, the `b-val` must have a value of "0".

M	A	DMS +PU+	M-DMS	MIU	n/a

For more information about the RTP media transport and the "limited random access data availability" model, see the following guidelines.

7.5.4.2.14 MT normative random access data availability models

7.5.4.2.15 MT "full random access data availability" model

7.5.8.2.39 MT RTP receiving endpoint range header

7.5.8.2.40 MT RTP serving endpoint range header

Excluding variants related to printing, the following examples apply to this guideline: [C1] and [C2].

7.4.1.3.20.2 If the serving endpoint assigns "0" to both `a-val` and `b-val` of the `op-param`, then the `op-param` must be omitted from the 4th field.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	[39] [51] [52]

The following examples apply to this guideline: [C1] and [C2].

7.4.1.3.21 MM ps-param (server-side playspeeds parameter)

7.4.1.3.21.1 The definition of `ps-param` must be as follows.

- `ps-param` = [`ps-param-delim`] "DLNA.ORG_PS=" `ps-value`;
- `ps-param-delim` = ";";
- `ps-value` = [`server-speed` *("," `server-speed`)];

- server-speed = <conforms to the TransportPlaySpeed string, as specified in the AVTransport specification>.

The ps-value must be a comma-delimited list of play speed values. The ps-value must exclude the play speed of "1" from its list. If the media transport component (either for a server, client) does not support additional server-side play speeds beyond "1" for the context of the protocolInfo, then the fourth field must omit the ps-param (i.e. "DLNA.ORG_PS=1" is prohibited).

The format of each play speed value must conform to the TransportPlaySpeed string, as specified in 2.2.8 of [50].

If used in conjunction with a protocolInfo indicating "http-get" in the first field, then the use of the PlaySpeed.dlna.org HTTP header applies to the context of the protocolInfo. See 7.5.5.3.16 MT HTTP PlaySpeed.dlna.org Header for more information.

If used in conjunction with a protocolInfo indicating "rtsp-rtp-udp" in the first field, then the use of the SCALE RTSP header applies to the context of the protocolInfo. See 7.5.8.2.41 MT RTP scale header for more information.

If using the 4th_field syntax defined in 7.4.1.3.16.1 or 7.4.1.3.16.2 and if ps-param follows another parameter, then the ps-param must include the ps-param-delim.

M	A	DMP DMS DMC DMR +DN+ +UP+ +PU+ +PR2+	M-DMP M-DMS M-DMC M-DMS M-DMC M-DMU	MIU	[50] [51] [52]

This guideline defines the DLNA.ORG_PS parameter. The parameter indicates the transport layer's supported play speeds.

Excluding variants related to printing, the following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4]. The ps-param is not used for image content nor XHTML print documents, but control points of printing-related endpoints still need to tolerate the presence of the ps-param.

7.4.1.3.21.2 If the ps-param token appears in a res@protocolInfo or a protocolInfo of a UPnP AV connection, and if the first field of protocolInfo is "http-get", then the content source must be capable of responding (with a target response) to requests that indicate a valid and supported play-speed for the content binary (i.e., the associated HTTP server endpoint does not return error code 406 (not acceptable), unless other HTTP headers are the cause of the error.)

M	C	DMS +PU+	M-DMS	MIU	[50] [51] [52]

When used with a <res> element or UPnP AV connection, the DLNA.ORG_PS parameter identifies the server supports one or more optional server-side play speed operations.

Guideline 7.5.5.3.16 MT HTTP PlaySpeed.dlna.org header gives more information about responding to HTTP requests that use the PlaySpeed.dlna.org HTTP header.

Variants of the [C1] and [C2] examples that involve sources of audio and/or AV content apply to this guideline.

7.4.1.3.21.3 In conjunction with the rules defined in 7.4.1.3.21.1 and 7.4.1.3.21.2, the fourth field of a res@protocolInfo may use the ps-param for non-DLNA media format profiles.

O	A	DMP DMS DMC DMR +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This guideline permits the use of DLNA.ORG_PS for both DLNA and non-DLNA content. This guideline also permits the parameter to be used for HTTP and RTP and other transports not specified by DLNA.

The following examples apply to this guideline row: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4], except the printing variant for [C1]. Although the printing and download usages do not employ play speeds to achieve the system usage in a normative way, control points that parse 4th field parameters are required to tolerate the presence of the ps-param.

It should be noted that the CDS:CreateObject syntax has restrictions on the 4th field syntax, as described in 7.4.1.7.19.

7.4.1.3.22 MM ci-param (conversion indicator flag)

7.4.1.3.22.1 The syntax definition of ci-param must be as follows:

- ci-param = [ci-param-delim] "DLNA.ORG_CI=" ci-value;
- ci-param-delim = ";";
- ci-value = boolean;
- boolean = "1" | "0".

If the context of the protocolInfo involves a content binary that is converted from a different content binary, then ci-value is "1"., the ci-value is "0" (see 7.4.1.3.12 for determining an appropriate context).

If using the 4th_field syntax defined in 7.4.1.3.16.1 or 7.4.1.3.16.2 and if ci-param follows another parameter, then the ci-param must include the ci-param-delim.

M	A	DMP DMS DMC DMR DMR +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

The ci-param is a "conversion indication parameter".

An MSCP uses this parameter to select the most relevant resource from the available resources of a CDS object. If "1" is specified for this parameter value, then the resource is converted from a different content binary. Converted content usually has equal or worse quality.

Examples of conversion include transcoding, system layer conversion, scaling, and decoding.

This guideline also applies in upload AnyContainer and optional content management (OCM) operations. In those cases, a control point is responsible for setting this parameter when it knows the content related to the operation is a converted content binary. This applies for content that is intended for a content transfer process, as well as operations that involve a URI specified by the control point.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [C5], [P1], [P2], [P3], [P4], and [P5].

7.4.1.3.22.2 In conjunction with the rules defined in 7.4.1.3.22.1, the 4th field of a res@protocollInfo may use ci-param for the following scenarios:

- the content is conformant to a DLNA media format profile;
- the content is conformant to a non-DLNA media format profile;
- the first field of protocollInfo is "http-get" or "rtsp-rtp-udp";
- the first field of protocollInfo is not "http-get" or "rtsp-rtp-udp".

O	A	DMP DMS DMC DMR DMP+ +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This guideline permits the use of DLNA.ORG_CI for both DLNA and non-DLNA content. This guideline also permits the parameter to be used for HTTP and RTP and other transports not specified by DLNA.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [C5], [P1], [P2], [P3], [P4], and [P5].

7.4.1.3.22.3 If the UPnP MediaServer knows that the content of a <res> element is a converted content binary, then the MediaServer should use the ci-param in the protocollInfo value.

S	A	DMS	M-DMS	MIU	[52]

The ci-param is not mandatory, but its use is strongly recommended, especially for converted content.

How a content source knows if content is converted is outside the scope for the guidelines. In some cases, the content source knows because it is the entity that actually converts the content. In other cases, the content source might know because the content source was informed in an implementation-specific manner.

This guideline applies to MediaServers because of examples [C1] and [C2].

7.4.1.3.22.4 The ci-param may be omitted.

O	A	DMP DMS DMC DMR DMP+ +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[52]

This following examples apply to this guideline: [C1], [C2], [C3], [C4], [C5], [P1], [P2], [P3], [P4], and [P5].

7.4.1.3.22.5 If a protocollInfo value omits the ci-param, then UPnP MediaServer control points must infer that the associated content is not converted content.

M	A	DMP DMC DMR DMP+ +DN+ +UP+ +PU+ +PR2+	M-DMP M-DMD M- DMC M-DMU	MIU	[52]

The following examples apply to this guideline: [P1], [P2], [P3], [P4].

7.4.1.3.23 MM flags-param (flags parameter)

7.4.1.3.23.1 The syntax definition of the flags-param must be as follows:

- flags-param = [flags-param-delim] "DLNA.ORG_FLAGS=" flags-value;
- flags-param-delim = ";;";
- flags-value = primary-flags reserved-data;
- primary-flags = 8 hexdigit;
- reserved-data = 24 reserved-hexdigit;
- hexdigit = <hexadecimal digit: "0"-"9", "A"-"F", "a"-"f">;
- reserved-hexdigit = "0".

If the protocolInfo value omits the flags-param, then the default meaning for individual flags and values embedded in flags-param is determined by default value policies or is unknown (in the case where default values are not defined). As new flags and values are defined for flags-param, those guidelines will clarify the meaning for when flags-param is omitted or when "0" is used (see 7.4.1.3.23.2 for the default value policies).

Example

- DLNA.ORG_FLAGS=03100000000000000000000000000000

If using the 4th field syntax defined in 7.4.1.3.16.1 or 7.4.1.3.16.2 and if flags-param follows another parameter, then the flags-param must include the flags-param-delim.

M	A	DMP DMS DMC DMR DMR +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMR M-DMR M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

Many DLNA binary-value flags that belong in the 4th field are encapsulated in this parameter. This helps reduce the length of the 4th field as the number of binary-value parameters increases in the future. In a simple usage, a single bit in the binary representation of flags-value maps to a single binary-value parameter. In some cases, DLNA may choose to define that a series of bits represents a small integer.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4].

7.4.1.3.23.2 The primary-flags token must be exactly 8 hexadecimal digits and it must represent a value composed of 32 binary bits. Each bit must represent a binary flag. The least significant bit corresponds to bit-0 and the most significant bit corresponds to bit-31 (for example, 10000000000000000000000000000000b = 0x80000000 where bit-31 is the only bit set to true)

The bit mapping of primary-flags must be as follows:

- Bit-31: sp-flag (sender paced flag)
 - a) Applies to all DLNA transport protocols.
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have a value of unknown because sender-paced content and non-sender paced content is permitted by previous versions of the DLNA guidelines. Content receivers that fail to transfer and/or render content because they use transport flow control mechanisms are in violation of this guideline.
 - c) See the following for more information.
 - 7.4.1.3.27 MM sp-flag (sender paced flag)

- Bit-30: lop-npt (limited operations flags: time-based seek)
 - a) Applies to all DLNA transport protocols.
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an inferred value of false.
 - c) See the following for more information.
 - 7.4.1.3.28 MM lop-npt and lop-bytes (limited operations flags) – Common
- Bit-29: lop-bytes (limited operations flags: byte-based seek)
 - a) Applies to all DLNA transport protocols.
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an inferred value of false.
 - c) See the following for more information.
 - 7.4.1.3.28 MM lop-npt and lop-bytes (limited operations flags) – Common
- Bit-28: playcontainer-param (DLNA play container flag)
 - a) Applies to all DLNA transport protocols.
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an inferred value of false because this flag applies only to DMR devices and the DLNA PlayContainer URI operation is optional for DMR devices.
 - c) See the following for more information
 - 7.4.1.3.31 MM playcontainer-param (DLNA PlayContainer flag)
 - 7.4.1.4.25 MM rendering media collection
 - 7.4.1.4.28 MM DLNA PlayContainer URI
 - 7.4.1.4.29 MM control point rules for DLNA PlayContainer UR
- Bit 27: s₀-increasing (UCDAM s₀ increasing flag)
 - a) Applies to all DLNA transport protocols.
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then the s₀-increasing flag must have an inferred value of unknown. The previous guidelines do permit s₀-increasing behaviour, but the previous version of the DLNA guidelines (i.e. v1.0) do not define normative rules for using the Seek media operation (or the range or TimeSeekRange.dlna.org headers) with such content.
 - c) See the following for more information.
 - 7.4.1.3.32 MM s₀-increasing (UCDAM s₀ increasing flag)
- Bit 26: s_N-increasing (UCDAM s_N increasing flag)
 - a) Applies to all DLNA transport protocols.
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an inferred value of unknown because previous versions of the DLNA guidelines permit content that grows with time or has a fixed ending.
 - c) See the following for more information.
 - 7.4.1.3.33 MM s_N-increasing (UCDAM s_N increasing flag)
- Bit-25: rtsp-pause (pause media operation support for RTP serving endpoints)
 - a) Applies only to RTP media transport
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an inferred value of false because previous versions of the DLNA guidelines do not support the RTP media transport.
- Bit 24: tm-s (streaming mode flag)
 - a) Applies to all DLNA transport protocols.
 - b) AV and audio media class content must set at least the tm-s flag equal to true.

- c) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an inferred value of true only for audio-only and AV content. For all other content, the inferred value is false.
 - d) See the following for more information.
 - 7.4.1.3.30 MM tm-s (streaming mode transfer flag)
- Bit 23: tm-i (interactive mode flag)
 - a) Applies to all DLNA transport protocols.
 - b) Image media class content must set at least the tm-i flag equal to true.
 - c) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an inferred value of true only for Image content, XHTML print documents, and media collection files. For all other content, the inferred value must be false.
 - d) See the following for more information.
 - 7.4.1.3.35 MM tm-i (interactive mode transfer flag)
- Bit 22: tm-b (background mode flag)
 - a) Applies to all DLNA transport protocols, except RTP.
 - b) If the flags-param is omitted or the dlna-v1.5-flag is false, then this flag must have an unknown value.
 - c) See the following for more information.
 - 7.4.1.3.36 MM tm-b (background mode transfer flag)
- Bit 21: http-stalling (HTTP connection stalling flag)
 - a) Applies only to the HTTP media transport.
 - b) If the flags-param is omitted, then this flag must have an inferred value of false.
 - c) See the following for more information.
 - 7.4.1.3.37 MM http-stalling (HTTP connection stalling flag)
 - 7.5.5.3.6 MT HTTP pause/pause-release media operation: connection stalling method
- Bit 20: dlna-v1.5-flag (DLNA v1.5 versioning flag)
 - a) Applies to all DLNA transport protocols.
 - b) If the flags-param is omitted, then this flag must have an inferred value of false.
 - c) See the following for more information.
 - 7.4.1.3.24 MM dlna-v1.5-flag (DLNA v1.5 version flag)

All other bits in primary flags are reserved for future use and must have a value of false.

M	C	DMP DMS DMC DMR DMP+ +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

The first 8 hexadecimal digits represent 32 binary flags. DLNA defines meaning for some of these bits. Other bits are reserved for future definition and are required to have a value of false at this time.

It should be noted that the usages of defined bits are defined in other DLNA guidelines.

When a protocolInfo represents the capabilities of a content binary, the bits are intended to be a representation of the applicability to the content binary, not on the current conditions of the network or the server's ability to stream data at the current time. When a stream is attempted on a content binary with the tm-s flag set, the DMS is still free to return an error that the stream cannot be completed at the current time due to internal conditions.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4]. In some cases, the flags-param will provide information that is of no use to certain endpoints. However, endpoints that parse the 4th field are required to tolerate the presence of the flags-param.

7.4.1.3.23.3 The reserved-data must be exactly 24 hexadecimal digits. These hexadecimal digits are reserved for future use and must have a value "0".

M	C	DMP DMS DMC DMR +DMPr +DN+ +UP+ +PU+ +PR1+ +PR2	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

The first 8 hexadecimal digits are used for primary-flags and the rest are reserved and undefined at this time.

When DLNA defines new normative flags and/or parameters for the 4th field, those flags and parameters should be defined here. The alternative of declaring a new 4th field parameter is not recommended for future authors of DLNA guidelines.

In general, definitions of new flags or parameters (that will occupy the current space allocated for reserved-bytes) need to define a syntax and semantics for DMS, DMR, DMC, and push controllers. Furthermore, parameter values that have short hexadecimal representations (such as new binary flags) should occupy hexadecimal digits that are more significant while parameter values that have longer hexadecimal representations occupy the less significant hexadecimal digits.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4].

7.4.1.3.23.4 UPnP AV MediaServer control points must be tolerant of reserved-data tokens that do not have "0" value hexadecimal digits.

M	C	DMP DMC DMR DMPr +DN+ +UP+ +PU+ +PR2+	M-DMP M-DMD M- DMC M-DMU	MIU	[51] [52]

The following examples apply to this guideline: [P1], [P2], [P3], and [P4].

7.4.1.3.24 MM dlna-v1.5-flag (DLNAv1.5 version flag)

If the dlna-v1.5-flag of the primary-flags token is true, then it must mean the following:

- Bits [31,21] (inclusive) of the primary-flags token are valid for use (i.e. those bits are valid for use);
- Bits [19,0] of the primary-flags token have undefined values.

If the dlna-v1.5-flag of the primary-flags token is false, then it must mean the following:

- only Bit-21 of the primary-flags is defined for use;
- all other bits in the primary-flags token have inferred values as described in 7.4.1.3.23.2.

M	A	DMS +PU+	M-DMS	MIU	n/a

Bits [31,22] are not defined for previous versions of the DLNA guidelines.

Bit-21 is the only bit defined in an errata for the DLNA v1.0 guidelines.

7.4.1.3.25 MM maxsp-param (maximum RTSP speed header value)

The definition of maxsp-param must be as follows:

- maxsp-param = [maxsp-param-delim] "DLNA.ORG_MAXSP=" maxsp-param-value;
- maxsp-param-delim = ";";
- maxsp-param-value = 1*DIGIT ["." *DIGIT].

The value of maxsp-param-value must be greater than, or equal to, 1.

If maxsp-param is specified, then the RTP serving endpoint must support the speed header in a RTSP PLAY request if the value of the "speed" header is less than, or equal to, the attribute value of maxsp-param. The RTSP "speed" header is defined in RFC 2326 ([33]), 12.35.

If using the 4th field syntax defined in 7.4.1.3.16.1 or 7.4.1.3.16.2 and if maxsp-param follows another parameter, then the maxsp-param must include the maxsp-param-delim.

M	A	DMP DMS DMC DMR +DN+ +UP+ +PU+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52] [33]

Excluding the variants that involve printing and download, the following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4]. Endpoints that parse 4th field parameters need to tolerate the presence of this parameter.

7.4.1.3.26 MM other-param (vendor-defined 4th field parameters)

7.4.1.3.26.1 The definition of other-param is as follows:

- other-param = [other-param-delim] IANA-name "_" other-param-name "=" other-param-value;
- other-param-delim = ";";
- IANA-name = <IANA-registered name, with top level domain (e.g..net,.org,.com)>;
- other-param-name = *<"a"-"z", "A"-"Z", "0"-"9">;
- other-param-value = *<"a"-"z", "A"-"Z", "0"-"9", "_", ",", "+", "-">.

If using the 4th field syntax defined in 7.4.1.3.16.1 or 7.4.1.3.16.2 and if other-param follows another parameter, then the other-param must include the other-param-delim.

M	A	DMP DMS DMC DMR DMP+ +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This defines the syntax for vendor extensions in the 4th field of protocolInfo values.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4].

7.4.1.3.26.2 The industry may use other-param for vendor-specific parameters in the 4th field of a protocolInfo value.

O	A	DMP DMS DMC DMR DMP+ +DN+ +UP+ +PU+ +PR1+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

This guideline permits the use of vendor extensions in the 4th field.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [C5], [P1], [P2], [P3], and [P4].

7.4.1.3.26.3 A UPnP AV MediaServer that receives a CDS:CreateObject request to create a <res> element with a res@protocolInfo that has optional and/or vendor-defined 4th field parameters and if the MediaServer returns a success response, then the MediaServer must omit the unsupported 4th field parameters from the created <res> element.

M	A	DMS	M-DMS	MIU	[51] [52]

DMS devices are not required to interpret, understand, store, or maintain vendor-defined parameters in the 4th field.

This guideline is an extension of guideline 7.4.1.7.24.5, which requires a DMS to return only the supported metadata properties in a success response.

The following examples apply to this guideline: [P5]. As a corollary, control points involved in example [C5] are expected to tolerate responses that omit unsupported 4th field parameters.

7.4.1.3.27 MM sp-flag (sender paced flag)

7.4.1.3.27.1 The sp-flag (sender paced flag) indicates if the content source will act as the clock source, for the context of the protocolInfo.

- False = the content source is not the content clock source
- True = the content source is the content clock source

M	C	DMP DMS DMC DMR +DN+ +UP+ +PU+ +PR2+	M-DMP M-DMD M- DMS M-DMC M- DMU	MIU	[50] [51] [52]

The sender paced flag provides a way for the content source to indicate that it will send packets at the rate of the normal clock source.

In normal HTTP operation the content receiver endpoint is the source for the playback clock which controls the pace of the rendering. The content receiver endpoint uses TCP flow control to match the pace of the transfer of content to the pace of the playback. In some cases, the content source may be the content clock source, such as in the case of live broadcast content. This means that if the actual throughput (including any transmission delays caused by additional transmission loads on the network) is not sufficient for the content clock source, then the content source will take steps to ensure that the transmitted content binary matches the indicated media format profile, but the bitstream may show discontinuities through things like dropped frames.

Likewise, in RTP the serving endpoint is typically the clock source and controls the rate at which the content is sent to the network.

The following examples apply to this guideline: [C1], [C2], [C3], [C4], [P1], [P2], [P3], and [P4]. DMP+ and +PR1+ are excluded from this guideline because images do not have a clock source. However +PR2+ is required to tolerate the sp-flag when browsing a MediaServer.

7.4.1.3.27.2 In the context of SourceProtocolInfo values obtained from CMS:GetProtocolInfo for a UPnP AV MediaServer, the sp-flag must mean the following.

- False = the content source is never the content clock source for a protocolInfo when there is a “match by protocolInfo format”.
- True = the content source is capable of being the content clock source for a protocolInfo when there is a “match by protocolInfo format”.

M	C	DMS	M-DMS	MIU	[51] [52]

Variants of the [C1] and [C2] examples that involve MediaServers apply to this guideline.

7.4.1.3.27.3 In the context of SinkProtocolInfo values obtained from CMS:GetProtocolInfo for a UPnP AV MediaRenderer, the sp-flag must always be true or the flags-param must be omitted.

M	L	DMR	n/a	MIU	[50] [51]

Like DMP devices, DMR devices are required to support rendering of content, regardless of whether the content clock source is determined by the content source.

7.4.1.3.28 MM lop-npt and lop-bytes (limited operations flags) – Common

7.4.1.3.28.1 If the flags-param is present and either the lop-npt or lop-bytes bits are true, then the “limited random access data availability” model must be the data access model that applies in the context of the protocolInfo value and the op-param must be omitted from the 4th field of the protocolInfo value.

Specifically, this means that the transport operation (that is indicated by the lop-npt or lop-bytes) must be supported for a limited data range for the context of the protocolInfo, as defined in 7.5.4.2.16 MT “limited random access data availability” model.

The meaning of the lop-npt bit and lop-bytes flags are described in these guidelines, depending on whether the context is for the HTTP media transport or the RTP media transport.

7.4.1.3.29 MM lop-npt and lop-bytes (limited operations flags) – HTTP

7.4.1.3.30 MM lop-npt and lop-bytes (limited operations flags) – RTP

M	A	DMS +PU+	M-DMS	MIU	[51] [52]

The lop-npt indicates that the transport layer supports random access on a limited range of npt playback positions. Likewise, the lop-bytes indicates that the transport layer supports random access on a limited range of byte positions.

A fundamental difference between lop-npt/lop-bytes and the op-param is that lop-npt/lop-bytes assumes that s_0 can change, while op-param only permits changes to s_N . For additional information on the assumptions for the “limited random access data availability” model, see 7.5.4.2.16 MT “limited random access data availability” model.

The following examples apply to this guideline: [C1] and [C2]. +PR1+ does not apply to this guideline because it will never serve images under the “limited random access data availability” model.

7.4.1.3.28.2 Content receiver endpoints and UPnP AV MediaServer control points that attempt to acquire content data from the limited random access data range (defined in 7.5.4.2.16) MT “limited random access data availability” model must be able to properly request a valid range, even if the limited data range continuously changes with time.

This guideline only applies when “limited random access data availability” applies to the scenario.

M	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[50] [51] [52]

Content receiver endpoints and control points cannot assume anything special about the data range because the absolute beginning can change and the content may have no end (i.e., live content). See the following for more general information and examples of how this guideline applies to HTTP.

7.5.4.2.16 MT “limited random access data availability” model

7.5.5.2.20 MT HTTP: data range of “limited random access data availability”, requirement 7.5.5.2.20.2 and requirement 7.5.5.2.20.10

The [P1], [P2], and [P3] examples apply to the content receivers (excluding DMPr) that are governed by this guideline. DMPr devices are excluded because images and XHTML print documents are never transferred under the “limited random access data availability” model.

7.4.1.3.29 MM lop-npt and lop-bytes (limited operations flags) – HTTP

If the 4th field is associated with the “http-get” transport protocol, then the lop-npt and lop-bytes bits mean the following.

- lop-npt: indicates support of the TimeSeekRange.dlna.org HTTP header for the context of the protocolInfo under the “Limited Random Access Data Availability” model.
- lop-bytes: indicates support of the Range HTTP header for the context of the protocolInfo under the “limited random access data availability” model.

M	A	DMS +PU+	M-DMS	MIU	[39] [51] [52]

This guideline defines the lop-npt and lop-bytes bits, when HTTP is the transport protocol.

When used with a context involving HTTP, these bits identify if the server supports the TimeSeekRange.dlna.org or range HTTP headers for the associated content binary under the “limited random access data availability” model.

For more information on the “limited random access data availability” model for HTTP and these HTTP headers, see the following guidelines.

7.5.4.2.16 MT “limited random access data availability” model

7.5.5.2.18 MT HTTP common random access data availability requirements

7.5.5.2.20 MT HTTP: data of “limited random access data availability”

The following examples apply to this guideline: [C1] and [C2]. +PR1+ does not apply to this guideline because it will never serve images using the “limited random access data availability” model.

7.4.1.3.30 MM lop-npt and lop-bytes (limited operations flags) – RTP

If the 4th field is associated with the “rtsp-rtp-udp” transport protocol, then the lop-npt and lop-bytes bits mean the following.

- lop-npt: indicates support of the range header for the context of the protocolInfo under the “limited random access data availability” model.

- **lop-bytes:** In scenarios involving the RTP media transport, the lop-bytes bit must be set to false.

M	A	DMS +PU+	M-DMS	MIU	[39] [51] [52]

This guideline defines the lop-npt and lop-bytes bits, when RTP is the transport protocol.

When used with a context involving RTP, these bits identify if the server supports the range header for the associated content binary under the "limited random access data availability" model.

For more information on the "limited random access data availability" model for RTP and these RTP headers, see the following guidelines.

7.5.4.2.16 MT "limited random access data valability" model

7.5.8.2.52 MT RTP current limited data range indication

The following examples apply to this guideline: [C1] and [C2]. +PR1+ does not apply to this guideline because it will never serve images using the RTP media transport.

7.4.1.3.31 MM playcontainer-param (DLNA PlayContainer flag)

7.4.1.3.31.1 The playcontainer-param flag indicates support for a DLNA PlayContainer URI operation. If the flag is true for a protocolInfo, then it means that the UPnP AV MediaRenderer can play that type of content in a DLNA PlayContainer URI operation.

The playcontainer-param flag must be false when the context of the protocolInfo involves a media collection binaries (for example DIDL_S and DIDL_V playlist files). Furthermore, when performing a DLNA PlayContainer URI, the MediaRenderer must not render media collection binaries when traversing the CDS hierarchy. It should be noted that this restriction against playing media collection binaries applies only to media collection binaries defined by the DLNA guidelines.

M	A	DMR DMC +PU+	M-DMC	MIU	[50] [51] [52]

A DLNA PlayContainer URI allows a control point to instruct a DMR to browse a DMS and play content from it.

The playcontainer-param is used on a per-profile basis. For example, if the protocolInfo for "http-get" and "MPEG2_PS_NTSC" has playcontainer-param set to true, then MPEG2_PS_NTSC content will be played in playcontainer operation.

Example [C4] also applies to this guideline.

This also guideline applies to DMR devices because they can expose protocolInfo that has the playcontainer-param set to true through CMS:GetProtocolInfo.

This guideline also applies to control points that invoke CMS:GetProtocolInfo on a DMR because they have to parse 4th field values.

Since the res@dlina:trackTotal attribute is not required, there is not a consistent way to represent the individual tracks of media collection binaries. Furthermore, DLNA has no interoperability guidelines for navigating the tracks within a media collection binary. Therefore, these guidelines prohibit playback of media collection binaries until a future set of DLNA guidelines can adequately address these issues.

7.4.1.3.31.2 In the context of the source argument's `protocollInfo` values obtained from `CMS:GetProtocollInfo` for a UPnP AV MediaServer, the `playcontainer-param` flag must always be false if the `flags-param` is included for a `protocollInfo` value.

Likewise, 4th field values provided by a content source must set the `playcontainer-param` flag to false if the `flags-param` is included in the `protocollInfo` value.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	[51]

The DMR device class is the only device class that sets the `playcontainer-param` to true.

The following examples apply to this guideline: [C1] and [C2].

7.4.1.3.32 MM s_0 -increasing (UCDAM s_0 increasing flag)

7.4.1.3.32.1 The s_0 -increasing (UCDAM s_0 increasing flag) indicates if the UCDAM s_0 boundary is increasing.

- True = The s_0 data boundary increases with time.
- False = The s_0 data boundary is fixed.

M	A	DMS +PU+	M-DMS	MIU	[50] [51] [52]

If true, then the content does not have a fixed beginning. Otherwise, the content does have a fixed beginning (i.e. `npt=0` and `byte-pos=0` map to the beginning). It should be noted that the s_0 data boundary can reset, as described in the comments 7.5.4.2.16 MT "limited random access data availability" model requirement 7.5.4.2.16.3

The [C1] and [C2] examples apply to this guideline, with exception of those involving +PR1+ because images do not have beginnings that change with time.

7.4.1.3.32.2 If the s_0 -increasing flag is true then the following must apply to the context of the `protocollInfo`:

- the `op-param` must be omitted;
- if `lop-npt` and `lop-bytes` are both false, then the following must also apply;
- the s_0 data boundary must map to a beginning that is not static;
- the data range of $[s_0, s_N]$ must map to the `npt` range of `[npt-start-time, npt-last-time]` and the byte range of `[first-byte-pos, last-byte-pos]`, where `npt-start-time` and `npt-last-time` are in units of `npt` and `first-byte-pos` and `last-byte-pos` are in units of bytes;
- there exists a "live position" that must be equal to the s_N data boundary;
- if the s_N data boundary is changing with time, then the "live position" must shift forward in real-time;
- if the content source receives a transport layer request that is not a random access request (for example, HTTP request that omits `range` and `TimeSeekRange.dlna.org`) then the content source must respond with content data from the "live point";
- if either `lop-npt` or `lop-bytes` is true, then the "limited random access data availability" model must apply. See 7.5.4.2.16 MT "limited random access data availability" model for more information.

The rules in this guideline must apply even in scenarios where the transport server does not support random access requests (for example, HTTP requests with `range` or `TimeSeekRange.dlna.org`) for a content binary.

M	A	DMS +PU+	M-DMS	MIU	[50] [51] [52]

When s_0 -increasing is true, then the content binary has a beginning that can change. Since it is possible to set the s_0 -increasing flag to true and not support random access requests, it is necessary for this guideline to be applied in all scenarios.

For HTTP, this means that omitting the range and TimeSeekRange.dlna.org headers in an HTTP GET request results in the HTTP server endpoint returning content data bytes from a "live position" (as described in 7.5.5.2.20.9)). Furthermore, npt=0 or byte-pos=0 has no meaning (as described by 7.5.5.2.20.16)).

If lop-npt or lop-bytes is true, then the access model is governed by what is returned by availableSeekRange.dlna.org (as described in 7.5.5.2.20 MT HTTP data range of "limited random access data availability").

The [C1] and [C2] examples apply to this guideline, with exception of those involving +PR1+ because images do not have beginnings that change with time.

7.4.1.3.32.3 If the s_0 -increasing flag is false, then the following must apply to the context of the protocolInfo:

- the s_0 data boundary must map to a fixed and non-changing beginning;
- the data range of $[s_0, s_N]$ must occupy an npt range of $[0, \text{npt-last-time}]$ and a byte range of $[0, \text{last-byte-pos}]$, where npt-last-time is in units of npt and last-byte-pos is in units of bytes;
- the content binary's zero position (i.e. npt-time=0 and byte-pos=0) must map to the UCDAM's data position of s_0 ;
- the last-byte-pos and npt-last-time must map to the UCDAM's s_N data position and the s_N data boundary must map to the end of the available content data.

This guideline must apply even in scenarios where the transport server does not support random access requests (for example, HTTP requests with range or TimeSeekRange.dlna.org) for a content binary.

M	A	DMS +PU+	M-DMS	MIU	n/a

When s_0 -increasing is false, then the content binary has a fixed beginning. Since it is possible to set the s_0 -increasing flag to false and not support random access requests, it is necessary for this guideline to be applied in all scenarios.

For HTTP, this guideline means that HTTP requests that omit the range and TimeSeekRange.dlna.org headers results in the HTTP server endpoint returning content data from the absolute beginning of the content (as described by 7.5.5.2.19 MT HTTP data range of "full random access data availability").

Note that these assumptions and the required behaviour are consistent with assumptions of previous versions of the DLNA guidelines.

Note that s_0 -increasing=false permits mutually exclusive use of either the op-param or the lop-npt/lop-bytes with values of true, while s_0 -increasing=true prohibits use of the op-param but allows use of lop-npt and lop-bytes.

The [C1] and [C2] examples apply to this guideline, with the exception of those involving +PR1+ because images do not have beginnings that change with time.

7.4.1.3.32.4 In the context of sink argument's protocolInfo values obtained from CMS:GetProtocolInfo for a UPnP AV MediaRenderer, the s_0 -increasing flag must always be true or the flags-param must be omitted.

M	L	DMR	n/a	MIU	[50] [51] [52]

Like DMP devices, DMR devices are required to support normal playback rendering of content, regardless of the server's buffering model.

7.4.1.3.33 MM s_N -increasing (UCDAM s_N increasing flag)

7.4.1.3.33.1 The s_N -increasing (UCDAM s_N Increasing Flag) indicates if the UCDAM s_N boundary is increasing.

- True = The s_N data boundary increases with time.
- False = The s_N data boundary is fixed.

This flag applies regardless of whether the "full random access data availability" or "limited random access data availability" model is being used.

M	A	DMS +PU+	M-DMS	MIU	[50] [51] [52]

If true, then the content does not have a fixed ending. Otherwise, the content has a fixed ending.

In conjunction with 7.4.1.3.32, it is possible to determine whether the server exhibits a growing or sliding buffering model.

The [C1] and [C2] examples apply to this guideline, with exception of those involving +PR1+ because images do not have endings that change with time.

7.4.1.3.33.2 In the context of SinkProtocolInfo values obtained from CMS:GetProtocolInfo for a UPnP AV MediaRenderer, the s_N -increasing flag must always be true or the flags-param must be omitted.

M	L	DMR	n/a	n/a	[50] [51]

Like DMP devices, DMR devices are required to support normal playback rendering of content, regardless of the server's buffering model.

7.4.1.3.34 MM tm-s (streaming mode transfer flag)

7.4.1.3.34.1 If the tm-s flag is true, then the associated media transport content source must be capable of supporting the streaming mode transfer for the context of protocolInfo.

M	A	DMS +PU+	M-DMS	MIU	n/a

See Table 22.

It should be noted that content sources may generate an error response if it does not have the resources to respond at the current time.

The tm-s flag is not equivalent to the sp-flag. When the tm-s flag is true, it means that the content source is able to transmit fast enough for immediate rendering. If the sp-flag is false and the sustained throughput is less than what is needed for immediate rendering, then the content source will preserve the content binary's bitstream because the content source does not act as the clock source.

When the sp-flag is also true, it means that the content source is also the clock source for the content, which means that the content source will take steps to ensure that the content binary meets the expectations of the media format profile, but the rendering stream may have discontinuities (such as dropped frames).

The [C1] and [C2] examples apply to this guideline, with exception of those involving +PR1+ because the streaming transfer mode has no meaning for images and XHTML print documents.

7.4.1.3.34.2 The tm-s flag must be set to true for protocolInfo values where the pn-param indicates a media format profile for audio-only or AV media class. Setting the tm-s flag to true for images, XHTML print documents, or media collection binaries is expressly prohibited.

This requirement does not apply in scenarios where a res@protocolInfo value exists for a <res> element that does not have a URI value. (for example, a CDS object that was created in an upload AnyContainer operation and has yet to receive the actual content binary).

M	A	DMS +PU+	M-DMS	MIU	n/a

This ensures backwards compatibility with earlier DMP devices that always assume that audio-only and AV content is available for streaming.

See 7.5.4.2.3.2 (MT transfer mode support) for information on the transfer modes that a content receivers can specify when issuing requests.

The [C1] and [C2] examples apply to this guideline, with exception of those involving +PR1+ because the streaming transfer mode has no meaning for images and XHTML print documents.

7.4.1.3.35 MM tm-i (interactive mode transfer flag)

7.4.1.3.35.1 If the tm-i flag is true, then the associated media transport content source must be capable of supporting the interactive mode transfer for the context of protocolInfo.

M	A	DMS +PR1+ +PU+	M-DMS	MIU	n/a

See Table 22 for more information about interactive mode transfer.

It could be noted that content sources may generate an error response if it does not have the resources to respond at the current time.

The [C1] and [C2] examples apply to this guideline.

7.4.1.3.35.2 The tm-i flag must be set to true for protocolInfo values where the pn-param indicates a media format profile for the Image media class, XHTML print documents, or a media collection binary. Setting the tm-i flag to true for audio-only or AV content is expressly prohibited.

This requirement does not apply in scenarios where a res@protocolInfo value exists for a <res> element that does not have a URI value (for example, a CDS object that was created in an upload AnyContainer operation and has yet to receive the actual content binary)

M	A	DMS +PR1+ +PU+	M-DMS	MIU	n/a

This ensures backwards compatibility with earlier DMP devices that always assume that image content is available for immediate rendering.

See guideline 7.5.4.2.3.2 (MT transfer mode support) for information on the transfer modes that a content receiver can specify when issuing requests.

The [C1] and [C2] examples apply to this guideline.

7.4.1.3.36 MM tm-b (background mode transfer flag)

If the tm-b flag is true, then the associated HTTP server must be capable of supporting the Background Mode Transfer for the context of the protocolInfo.

In the context of a UPnP AV connection, a res@protocolInfo value, or contentFeatures.dlna.org value, the following restrictions must also apply.

- If the http-stalling flag is true, then tm-b flag must be set to true.
- If the sp-flag is true, then tm-b flag must be false.

M	A	DMS +PR1+ +PU+	M-DMS	MIU	n/a

See Table 22 for more information about background mode transfer.

It should be noted that the content source may generate an error response if it does not have the resources to respond at the current time.

Unlike the tm-s and tm-i flags, the tm-b flag can be used with all media classes.

It should be noted that a server that supports the http-stalling flag must also support the tm-b flag because a device that supports indefinite stalling implicitly supports lower transmission throughputs that can result from actively managing TCP flow control for a background transfer. The converse is not true because the ability to support a background transfer does not necessarily imply the ability to support indefinite stalling via TCP flow control.

Also note that a background transfer cannot be used in conjunction with server-paced content. Lower transmission throughputs resulting from a background transfer can cause the server's buffer to overflow. Content receivers that want to download content that have sp-flag=true need to use the streaming download media operation.

See guideline 7.5.4.2.3.2 (MT transfer mode support) for information on the transfer modes that a content receiver can specify when issuing requests.

The [C1] and [C2] examples apply to this guideline.

7.4.1.3.37 MM http-stalling (HTTP connection stalling flag)

If the http-stalling flag is true, then the associated HTTP server must be capable of supporting the Connection Stalling method for the pause and pause-release media operations on the content binary.

M	A	DMS +PU+	M-DMS	MIU	n/a

The connection stalling is a mechanism where a content receiver and a content source cooperatively use standard TCP flow control to temporarily pause the transmission of data.

HTTP server endpoints are not to misinterpret HTTP-level transport inactivity as a symptom of a TCP disconnect because a properly stalled HTTP client endpoint will use standard TCP flow control to keep the TCP connection alive. HTTP server endpoints should also be careful to not overflow their local network buffers when the connection stalling method is being used.

This [C1] and [C2] examples apply to this guideline when the scenario involves audio or AV content.

7.4.1.3.38 MM MediaServer UPnP AV connection behaviour

A UPnP AV MediaServer must have a Connection ID "0" to represent all transport layer connections that cannot be mapped to a particular transport layer connection.

M	R	DMS	M-DMS	n/a	[51] [55]

A DMS that implements this type of behaviour uses connection ID "0" to represent one or more transport layer connections. This behaviour clarifies the connection assumptions for DMS-1. Specifically, DMS devices are required to have at least one UPnP AV connection with Connection ID of "0" to represent an unknown number of connections (zero or more) to content receiver endpoints.

7.4.1.3.39 MM inaccurate context of ConnectionID = 0

UPnP AV MediaServer control points must not rely on the accuracy of the protocolInfo and peer connection information from CMS:GetCurrentConnectionInfo for ConnectionID = "0" on a UPnP AV MediaServer.

M	A	DMP DMR DMC +DN+ +UP+	M-DMP M-DMC M- DMD M-DMU	MIU	[51]

In the case of the UPnP AV connection "0", the information is assumed to be inaccurate because the context of ConnectionID="0" represents numerous requests for different content.

7.4.1.3.40 MM UPnP AV connection ID and instance ID assignment rule

7.4.1.3.40.1 UPnP AV MediaServers may have UPnP AV connections exposed through CMS:GetCurrentConnectionIDs.

O	C	DMS	M-DMS	n/a	[51] [55]

Enumeration of UPnP AV connections is out-of-scope but it is not expressly prohibited. DLNA defines no interoperability rules for the enumeration of such connections.

7.4.1.3.40.2 A UPnP AV MediaRenderer must always have at least the UPnP AV connection with a ConnectionID value of "0".

M	A	DMR	n/a	n/a	[51] [54]

DMR devices are always required to have a UPnP AV connection with a connection ID of "0". This allows control points to determine if the DMR supports a virtual instances for AVTransport and RenderingControl services.

7.4.1.3.40.3 A UPnP AV device must use the virtual instance ID of "-1" to indicate that a virtual instance (for RenderingControl or AVTransport services) does not exist for a UPnP AV connection.

M	R	DMS DMR	M-DMS	n/a	[50] [51] [60]

This is normative, per the UPnP AV specifications. The CMS:PrepareForConnection and CMS:GetCurrentConnectionInfo actions return virtual instance ID values through the RcsID, AVTransportID output arguments.

7.4.1.3.40.4 If a UPnP AV MediaRenderer supports a virtual instance for the AVTransport service, then it must minimally support AVTransport requests that specify a virtual instance ID of "0".

M	L	DMR	n/a	n/a	[50] [51] [54]

This allows control points to assume that AvtID=0 is always available, when the service is supported.

7.4.1.3.40.5 If a UPnP AV MediaRenderer supports a virtual instance for the RenderingControl service, then it must minimally support RenderingControl requests that specify a virtual instance ID of "0".

M	L	DMR	n/a	n/a	[51] [54] [60]

This allows control points to assume that RcsID=0 is always available, when the service is supported.

7.4.1.4 MediaServer requirements

7.4.1.4.1 MM ObjectID usage

7.4.1.4.1.1 UPnP AV MediaServers must assign a unique object ID for each entry in their ContentDirectory service (CDS). This rule applies to both container and item objects in a CDS metadata hierarchy.

M	R	DMS	M-DMS	n/a	[52]

This is a requirement of the CDS specification. This guideline's scope for uniqueness is for the entire CDS hierarchy.

7.4.1.4.1.2 UPnP AV MediaServers should maintain the object ID value on a persistent basis.

S	L	DMS	M-DMS	n/a	[52]

The purpose of this recommendation is to allow control points to implement features like "my favorite content". Although control points cannot assume that object ID values are persisted, this recommendation allows a control point to easily check if a CDS object is still available on the network.

The reason why this is not mandatory is that some embedded devices may have difficulty persisting object ID values.

7.4.1.4.1.3 A UTF-8 encoded object ID must not exceed 256 bytes in the XML escaped form encoded in UTF-8.

M	L	DMS	M-DMS	n/a	[52]

Provides a reasonable maximum length for object ID values, which are essential for CDS object declarations.

This guideline only applies to creation of object ID values.

7.4.1.4.1.4 DIDL-Lite documents or fragments that contain one or more CDS objects (i.e. <item> or <container> element) must have a unique value for each object's @id attribute.

M	R	DMS	M-DMS	n/a	[52]

This is a requirement of the CDS specification. This guideline's scope for uniqueness is limited to the DIDL-Lite document or fragment. For example, a UPnP AV MediaRenderer that reports metadata for a media collection cannot use the same object ID for each of the items in the media collection.

7.4.1.4.2 MM CDS – Browse unsorted order

If a UPnP AV MediaServer responds to a CDS:Browse request that specifies *BrowseFlag*=BrowseDirectChildren and an empty *SortCriteria* argument, then the MediaServer must preserve the indexed order of returned CDS objects for a given UpdateID output argument value.

As required by 2.5.11 in [52], the value of UpdateID must either be the CDS.SystemUpdateID value or a ContainerUpdateID. (Vendors are permitted to use either one for any given container.)

If returning a ContainerUpdateID, the mandatory portions for the "ContainerUpdateID" entry (in Table 1, 2.3 of [52]) must be implemented. Specifically, the ContainerUpdateID must change only when the browsed container is modified, which is also defined in the "container modification" entry in Table 1 of [52].

If returning the CDS.SystemUpdateID value, the mandatory portion of 2.5.20 in [52] must be implemented. Specifically, the CDS.SystemUpdateID must change whenever anything in the CDS changes.

MediaServer implementations that always return a different value for the UpdateID violate this guideline because it prevents incremental browsing behaviours that are necessary for DLNA MediaServer control points.

M	C	DMS	M-DMS	n/a	[52]

This requirement is implied in the CDS specification because the *StartingIndex* and *RequestedCount* input arguments are designed for incrementally a browsing a CDS container. The only time when the unsorted order may be different between two CDS:Browse requests will be when the UpdateID output argument values are different in the two responses. This is because the UpdateID value is a state counter for the CDS container or the entire CDS hierarchy.

Note that a MediaServer is permitted to always return the UpdateID output argument equal to CDS.SystemUpdateID. This guideline does not require MediaServer implementations to maintain internal state for ContainerUpdateIDs. Whether the MediaServer returns a ContainerUpdateID or the CDS.SystemUpdateID for any given CDS container is up to the vendor's decision.

7.4.1.4.3 MM DIDL-Lite multiple res – Formats

A CDS object (identified through an <item> or <container> element) should use multiple <res> elements (instead of multiple CDS objects) when the <res> elements represent the same content in different media format profiles.

S	R	DMS	M-DMS	n/a	[52]

These guidelines recommend a UPnP AV MediaServer device to expose multiple <res> elements for a single CDS object. This allows a UPnP AV MediaServer control point to present a single CDS object to the user, without necessarily presenting each variant of the same content.

This requirement applies when the DMS can determine that content binaries are the same content. The following are some examples of how multiple <res> elements can be used:

- the DMS acquired the content binaries in such a way that it knows that they are the same content;
- the DMS has transcoded locally stored content to other formats;
- the DMS advertises tuner-sourced content in multiple media format profiles;
- the DMS has a single, locally stored file that is advertised with multiple media format profiles, without any conversions;
- the DMS advertise the same video or image content (even when the media format profile is the same) but in different resolutions.

This requirement does not require a DMS to determine whether separate, locally stored files are actually the same content because this is difficult to do computationally and relying on embedded metadata is not always accurate. Furthermore, a DMS is not required to determine that content binaries uploaded through multiple upload AnyContainer or OCM: upload content operations requests are the same content.

This guideline is required because the MediaServer can always determine if it is providing a content transformation.

7.4.1.4.4 MM DIDL-Lite multiple res – Transports

A CDS object (identified through an <item> or <container> element) that has the same content available for different media transport protocols must expose the content through multiple <res> elements.

M	R	DMS	M-DMS	n/a	[52]

See the comments in 7.4.1.4.3 for more information.

7.4.1.4.5 MM DIDL-Lite content – Multiple points of accessibility

7.4.1.4.5.1 A UPnP AV MediaServer that does not receive the ALLIP value (case-sensitive) as part of the filter argument (in a CDS:Browse or CDS:Search request) must return only the URIs that are associated with (or treated as or assumed to be routable from) the network interface that received the SOAP request.

URIs with domain names may appear in these types of SOAP responses, according to the rules specified in 7.4.1.3.10.

This guideline applies to URIs in <res>.

This guideline applies to any URI value that uses an IPv4 network address, regardless of whether the content conforms to a DLNA media format profile.

M	L	DMS	M-DMS	n/a	[52]

These guidelines explain how a UPnP AV MediaServer is to handle the reporting of <res> elements when the UPnP AV MediaServer has multiple network interfaces.

Essentially, the default behaviour is that a UPnP AV MediaServer will only return <res> elements where the URIs are known (or assumed to be) routable from the network interface that received the request. However, if a UPnP AV MediaServer control point wants to receive URIs for all network interfaces, then the DMP can specify the ALLIP value as part of the filter argument. In such a scenario, a UPnP AV MediaServer is obligated to return all of the <res> elements for all of the active network interfaces that the UPnP AV MediaServer uses for media transport.

Because the guidelines language uses the UPnP AV MediaServer of a given UDN, a UPnP AV MediaServer device that uses a different UDN for each network interfaces (equivalent to multiple UPnP AV MediaServers for a single UPnP AV MediaServer device) does not need to return the <res> elements that are accessible on a different network interface (for example, res element found on a different logical UPnP AV MediaServer).

Annex C describes the subtleties of multiple network interfaces and the role of these guidelines in more detail.

7.4.1.4.5.2 A UPnP AV MediaServer that receives the ALLIP value (possibly with other filter values, including the star,* value) in the filter argument (of a CDS:Browse or CDS:Search request) must return all URIs associated with the UPnP AV MediaServer of a given UDN, regardless of whether the URI is thought to be routable from the network interface that received the SOAP request.

A UPnP AV MediaServer must expose all URI values either through multiple <res> elements (for each CDS object) or multiple CDS objects. See guidelines 7.4.1.4.5.3 and 7.4.1.4.5.4 for the description of how all URI values are exposed through multiple <res> elements or multiple CDS objects.

URIs with domain names may appear in these types of SOAP responses, according to the rules specified in 7.4.1.3.10.

This guideline applies to URIs in <res>.

This guideline applies to any URI value that uses an IPv4 network address, regardless of whether the content conforms to a DLNA media format profile.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.5.3 In conjunction with the guideline, a UPnP AV MediaServer that receives the ALLIP value should return CDS objects with multiple <res> elements, so that some of these <res> elements are URI values that point to the same content available on different network interfaces.

S	C	DMS	M-DMS	n/a	[52]

This guideline allows a UPnP AV MediaServer to report content availability on multiple networks through multiple <res> elements. The presence of multiple <res> elements is still governed by 7.4.1.4.5.2.

7.4.1.4.5.4 A UPnP AV MediaServer that does not receive the ALLIP value (possibly with other filter values, including the asterisk,*, value) may return CDS objects with zero or more <res> elements.

Note that it is generally true that a UPnP AV MediaServer may return CDS objects with zero or more <res> elements.

O	C	DMS	M-DMS	n/a	[52]

Although it is implied that a UPnP AV MediaServer is not required to provide a <res> element for a CDS object, this guideline states it explicitly. This guideline allows implementations that rely on multiple CDS objects (instead of multiple <res> elements to represent different versions of the same content) to comply with 7.4.1.4.5.1.

7.4.1.4.6 MM DIDL-Lite multiple res – Thumbnails

7.4.1.4.6.1 If a UPnP AV MediaServer exposes a CDS object with a <upnp:class> designation of object.item.imageItem (or any class derived from it), then the UPnP AV MediaServer should provide a <res> element for the thumbnail resource (multiple thumbnail <res> elements are also allowed).

S	C	DMS	M-DMS	n/a	[52]

UPnP AV MediaServer devices that implement thumbnail support reduce the network load for themselves and for control points that display thumbnails to the user.

7.4.1.4.6.2 If a UPnP AV MediaServer exposes a CDS object with a <upnp:class> designation of object.item.videoItem (or any class derived from it), then the UPnP AV MediaServer should provide a <res> element for the thumbnail resource (multiple thumbnail <res> elements are also allowed).

S	C	DMS	M-DMS	n/a	[52]

7.4.1.4.6.3 If a UPnP AV MediaServer exposes thumbnail images for image or video content, then a UPnP AV MediaServer must provide a thumbnail that conforms to requirement 7.1.5 JPEG TN Format Profile media format profile in IEC 62481-2 and be declared with the JPEG_TN designation in the fourth field of the res@protocolInfo attribute.

M	L	DMS	M-DMS	n/a	IEC 62481-2, [52]

When thumbnails are provided, the minimal expectation is to provide JPEG thumbnails. However, vendors may also provide additional thumbnails of other formats (such as a PNG thumbnail).

7.4.1.4.6.4 If a UPnP AV MediaServer exposes thumbnail images for image or video content, then a UPnP AV MediaServer may provide additional <res> elements for thumbnail images.

O	C	DMS	M-DMS	n/a	[52]

7.4.1.4.6.5 A UPnP A/V media server must not expose a <res> element with a thumbnail media format profile ID (i.e., JPEG_TN, PNG_TN), without exposing at least one additional <res> element with a media format profile ID that is not one of the thumbnail media format profile IDs.

M	R	DMS	M-DMS	n/a	[52]

Thumbnails are designed to augment other content items of any media class (audio, AV, or images) and are not meant to represent stand-alone images.

Images that will not be used as thumbnails but which match the thumbnail resolution should be exposed using the appropriate smallest image media format profile ID (for example, JPEG_SM).

7.4.1.4.7 MM DIDL-Lite audioItem album art

7.4.1.4.7.1 If a UPnP AV MediaServer exposes a CDS object with a <upnp:class> designation of object.item.audioItem or object.container.album.musicAlbum (or any class derived from either class), then the UPnP AV MediaServer should provide a <upnp:albumArtURI> element to present the URI for the album art.

S	C	DMS	M-DMS	n/a	[52]

Unlike image or video content, thumbnails for audio content should be presented through the <upnp:albumArtURI> element.

7.4.1.4.7.2 If a UPnP AV MediaServer exposes one or more <upnp:albumArtURI> elements for a single CDS object, then at least one of the URI values should point to thumbnail album art conforming to guideline 7.1.5 JPEG TN format profile for the JPEG_TN media format profile in IEC 62481-2.

S	C	DMS	M-DMS	n/a	IEC 62481-2, [52]

If album art thumbnails are provided, the desired expectation is to have JPEG thumbnails. Additional thumbnails can also be provided.

7.4.1.4.7.3 If a UPnP AV MediaServer exposes a <upnp:albumArtURI> element with a URI pointing to a thumbnail conforming to a DLNA media format profile, then the <upnp:albumArtURI> must have the albumArtURI@dlna:profileID attribute that identifies the DLNA profile ID of the thumbnail.

The namespace for DLNA defined properties must be "urn:schemas-dlna-org:metadata-1-0/" and the namespace prefix must be "dlna:".

The following is an example.

- <upnp:albumArtURI dlna:profileID="JPEG_TN" xmlns:dlna="urn:schemas-dlna-org:metadata-1-0/">http://192.168.1.1/album/albumArt1.jpg</upnp:albumArtURI>

M	A	DMS	M-DMS	n/a	[52]

This guideline allows control points, a convenient way to identify thumbnails that conform to a DLNA media format profile.

7.4.1.4.8 MM IFO file

7.4.1.4.8.1 If a UPnP AV MediaServer exposes a content binary profiled according to MPEG-2 AV format, usage of profile IDs, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.11 (MPEG_PS_NTSC or MPEG_PS_PAL profiles) in IEC 62481-2, then the UPnP AV MediaServer must either ensure that there are no SCR and/or PTS discontinuities (as defined in 7.4.1.4.8.6) or generate an IFO file if it detects discontinuity.

It should be noted that this guideline does not apply in the scenario where a DMS exposes a content binary that was imported by the DMS, except when such content is received from a DLNA source.

All guidelines in 7.4.1.4.8 MM IFO do not apply in scenarios involving URIs and/or <res> elements for RTP media transport because RTP encapsulation and padding make the offsets in the IFO file inaccurate.

M	A	DMS	M-DMS	n/a	IEC 62481-2, [52]

Some decoders cannot handle the SCR/PTS discontinuous PS stream without proper additional decoder-specific control. This guideline provides the method that allows the DMP to obtain the information about the SCR/PTS discontinuous regions in program stream-profiled content.

The following are examples of content where a UPnP AV media server must comply with this guideline when exposing such content:

- a DMS application running on an open platform, such as a PC, that records, generates, or edits content direct.

The following are examples of content where a UPnP AV media server does not have to comply with this guideline when exposing such content:

- a DMS application running on an open platform, such as a PC, that receives or copies content from a non-DLNA source (for example, Internet);
- a DMS application running on an open platform, such as a PC, which has other separate applications that record, create, or edit content or that import content from non-DLNA sources (for example, Internet).

Although the above comments give examples where this guideline does not apply, it is still strongly recommended that vendors provide IFO files in all cases when exposing an MPEG2 PS profiled content binary that has a discontinuity.

7.4.1.4.8.2 If a UPnP AV MediaServer exposes a content binary profiled according to MPEG-2 AV format, usage of profile IDs, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.11 (MPEG_PS_NTSC or MPEG_PS_PAL profiles) in IEC 62481-2, along with an associated IFO file, then it must also expose the IFO file through the res@dlina:ifoFileURI attribute to present the URI for the IFO file as defined in 7.4.1.4.8.7.

If a UPnP AV MediaRenderer control point serves a content binary, profiled according to MPEG-2 AV format, usage of profile IDs, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.11 (MPEG_PS_NTSC or MPEG_PS_PAL profiles) in IEC 62481-2, direct to a MediaRenderer, then the MediaRenderer control point must provide the IFO file at the transport layer in a manner consistent with 0 MT HTTP Header: dlina pragma-directive (ifoFileURI.dlina.org).

The namespace for DLNA defined properties must be "urn:schemas-dlna-org:metadata-1-0/" and the namespace prefix must be "dlina:".

M	A	DMS +PU+	M-DMS	n/a	[39] [52] [54]

7.4.1.4.8.3 If a UPnP AV MediaServer exposes a content binary profiled according to MPEG-2 AV format, usage of Profile IDs, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.11 (MPEG_PS_NTSC or MPEG_PS_PAL profiles) in IEC 62481-2 without SCR or PTS discontinuities, the UPnP AV MediaServer may provide a `res@_dlna:ifoFileURI` attribute to present the URI for the IFO file as defined in 7.4.1.4.8.7.

If a UPnP AV MediaRenderer control point serves a content binary, profiled according to MPEG-2 AV format, usage of profile IDs, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.11 (MPEG_PS_NTSC or MPEG_PS_PAL profiles) in IEC 62481-2 without SCR or PTS discontinuities, then the MediaRenderer control point may provide the IFO file at the transport layer in a manner consistent with 0 MT HTTP Header: `dlna pragma-directive (ifoFileURI.dlna.org)`.

The namespace for DLNA defined properties must be "urn:schemas-dlna-org:metadata-1-0/" and the namespace prefix must be "dlna:".

O	A	DMS +PU+	M-DMS	n/a	[39] [52] [54]

An IFO file may contain other metadata which may be useful to the rendering endpoint.

7.4.1.4.8.4 If an IFO file is provided for a MPEG_PS_NTSC or MPEG_PS_PAL profiled content binary via an associated `res@dlna:ifoFileURI` attribute and/or the transport layer (such as described in 7.5.5.2.11 MT HTTP header: `dlna pragma-directive (ifoFileURI.dlna.org)`), rendering endpoints must render the content item even if the MPEG stream contains discontinuous SCR and/or PTS.

M	A	DMP DMR	M-DMP	n/a	n/a

If a MediaServer does not provide an IFO file (for example, `res@dlna:ifoFileURI` is omitted) and a content binary, profiled according to MPEG-2 AV format, usage of profile IDs, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.11 in IEC 62481-2, has discontinuities, then the rendering endpoint may choose not to render the content binary.

7.4.1.4.8.5 If a rendering endpoint attempts to render a content binary that is profiled according to MPEG-2 AV format, usage of profile IDs, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.11 in IEC 62481-2 that has discontinuities and no IFO file is available, then the rendering endpoint must gracefully recover from the failure condition caused by any discontinuity.

M	A	DMP DMR	M-DMP	n/a	n/a

7.4.1.4.8.6 An SCR or PTS discontinuity is defined as the occurrence of one of the following conditions in an MPEG2 PS content binary.

Condition1:

- if $SCR(0) + SCRMaxValue - SCR(-1) \leq 0,7$ (to cover wrap-around case)
 $SCR(0) + SCRMaxValue < SCR(-1) + PackDuration$;
- otherwise $SCR(0) < SCR(-1) + PackDuration$.

or

Condition2:

- if $PTS(-1) > PTS(0)$ (to cover wrap-around case)
 $PTS(0) + PTSMaXValue - PTS(-1) > 0,61sec$;
- otherwise $PTS(0) - PTS(-1) > 0,61sec$.

where

- SCR(0) is the SCR of the current pack;
- SCR(-1) is the SCR of the preceding pack;
- SCRMaxValue is $2^{32} \cdot 300$;
- PackDuration is $\text{int}((\text{PackSize} \cdot 27000000)/(\text{program\_mux\_rate} \cdot 50)) = \text{int}((2048 \cdot 27000000)/(25200 \cdot 50)) = 43885$;
- PTS(0) is the PTS of the first picture of current GOP;
- PTS(-1) is the PTS of the first picture of preceding GOP;
- PTSMaxValue is 2^{32} .

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	ISO/IEC 13818-1, ISO/IEC 13818-9

In ISO/IEC 13818-1, 2.7.1, there is the following definition: "The Program Stream must be constructed so that the time interval between the bytes containing the last bit of system_clock_refrence_base fields in successive packs must be less than, or equal to, 0,7s."

It should be noted that SCR_base and PTS are limited to 32bit in the guideline.

It should be noted that all the mathematical equations on calculating SCR/PTS discontinuity should be performed using unsigned integer arithmetic as described in ISO/IEC 13818-1, 2.4.3.7.

7.4.1.4.8.7 If a UPnP AV MediaServer provides an IFO file, then the URI of the IFO file must be specified.

IFO file URIs are governed by guidelines in 7.4.1.3.10, except that the maximum length for an IFO file URI is 900 bytes.

Example:

- `<res protocolInfo="http-get:*:video/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC" duration="02:45:00" dlina:ifoFileURI="http://192.168.0.1:8080/IFO_101.ifo" xmlns:dlina="urn:schemas-dlna-org:metadata-1-0/">
http://192.168.0.1:8080/MPEG/ntsc001.mpg
</res>`

In this case, the URI of the IFO file is `http://192.168.0.1:8080/IFO_101.ifo`

M	A	DMS	M-DMS	n/a	[52]

Content receivers can get the IFO file by issuing HTTP GET requests using this URI.

The reason for a shorter IFO file URI is because they are included in HTTP headers where the length is constrained.

7.4.1.4.9 MM CDS browse/search action – Filter argument

7.4.1.4.9.1 The following five metadata properties must always be present in the DIDL-Lite response, even if the metadata properties are not specified in the filter argument of a CDS:Browse or CDS:Search request:

- @id;
- @parentID;
- @restricted;
- dc:title;

- upnp:class.

M	C	DMS	M-DMS	n/a	[52] [55]

The filter argument of the CDS:Browse and CDS:Search action instructs a UPnP AV MediaServer ContentDirectory to return only the specified metadata properties in the DIDL-Lite response of the result output argument. This guideline clarifies that some metadata properties are required to be present even if they are not specified in the filter argument.

7.4.1.4.9.2 If an element of metadata property is specified, then the required attributes of the metadata element must be presented.

For example:

- If a control point specifies "res" in the Filter, then res@protocolInfo is returned.

M	R	DMS	M-DMS	n/a	[52]

7.4.1.4.9.3 A UPnP AV MediaServer control point should explicitly specify the desired metadata properties in the Filter input argument of a CDS:Browse or CDS:Search request.

S	R	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52] [55]

This guideline recommends control points to limit the requested metadata to only the metadata that will be used by the control point. A filter value of asterisk "*" will likely cause the UPnP AV MediaServer to send more metadata than what the control point can actually use.

7.4.1.4.9.4 In conjunction with 7.4.1.4.9.1, a UPnP AV MediaServer device must not return metadata properties unless specified in the filter argument.

For example:

- if a control point does not specify res@importUri in the filter, then it is not returned.

It should be noted that having an attribute property in the filter automatically requires the MediaServer to return the element to which the attribute belongs.

For example:

- if the control point specifies res@importUri (without "res"), then the "res" is also returned.

M	R	DMS	M-DMS	n/a	[52] [55]

7.4.1.4.9.5 A UPnP MediaServer must return DLNA metadata (i.e., attributes or elements with the dlna: prefix) only when the filter argument indicates a request for the particular DLNA attribute(s) or element(s).

It should be note that "*" indicates a request for all attributes and elements.

M	C	DMS	M-DMS	n/a	[55]

This behaviour is required by the ContentDirectory specification and is consistent with 7.4.1.4.9.4.

Guidelines that require DLNA-defined metadata do not overrule underlying rules specified by the UPnP AV ContentDirectory service.

For example, guideline 7.4.1.4.7.3 requires albumArtURI@dlina:profileID, but a MediaServer only includes albumArtURI@dlina:profileID in a response when the filter argument indicates a request for it.

7.4.1.4.9.6 A UPnP MediaServer should declare the dlina: namespace (i.e. "urn:schemas-dlna-org:metadata-1-0/") only when the filter argument indicates a request for one or more DLNA attributes or elements and one or more DLNA attributes are included in the DIDL-Lite response.

It should be noted that "*" indicates a request for all attributes and elements.

S	C	DMS	M-DMS	n/a	[55]

This guideline reduces the computational requirements of control points that employ validating schemas.

Although some guidelines require DLNA-defined elements and attributes in certain situations, the DLNA schema does not play a syntax enforcement role as all DLNA-defined elements and attributes are considered optional from the schema's perspective.

For example, guideline 7.4.1.4.7.3 albumArtURI@dlina:profileID when the associated URI points to an image that is compliant to a DLNA media format profile. Since it is impossible for the DLNA schema to determine if a URI points to a DLNA media format profile, the albumArtURI@dlina:profileID is considered an optional attribute (from a schema perspective) even though it is required from a guidelines perspective.

7.4.1.4.10 MM CDS browse/search action – Reduced response behaviour

7.4.1.4.10.1 A UPnP AV MediaServer device may reduce the number of CDS objects (<item> and <container> elements) in a response to a CDS:Browse or CDS:Search for the following scenarios only:

- the transmission of a SOAP response with a huge byte length (>204 800 bytes);
- the transmission of a SOAP response that exceeds 30 s for the transmission time.

O	C	DMS	M-DMS	n/a	[52] [55]

This guideline allows a UPnP AV MediaServer to limit the number of CDS objects returned in the SOAP response, even if the control point specified a desire for more CDS objects in the RequestedCount input argument. The reason for permitting such behaviour is to allow UPnP AV MediaServer implementations to comply with other guidelines: 0 DDC UPnP SOAP packet size and 7.3.10 DDC UPnP device responsiveness.

7.4.1.4.10.2 The number of CDS object entries (total <item> and <container> elements) in the result output argument (containing the DIDL-Lite metadata) must match the value specified in the NumberReturned output argument.

M	C	DMS	M-DMS	n/a	[52] [55]

This guideline must be followed, even if a UPnP AV MediaServer reduces the number of CDS objects returned in the SOAP response.

7.4.1.4.10.3 If a UPnP AV MediaServer device reduces the number of CDS objects in a CDS:Browse(BrowseDirectChildren) or CDS:Search response then the number of returned CDS objects (as parsed in result) must be equal to the value of NumberReturned, which is less than RequestedCount.

M	C	DMS	M-DMS	n/a	[52] [55]

A UPnP AV MediaServer that limits the number of CDS objects is obligated to return a *NumberReturned* value that is consistent with the *RequestedCount* input argument.

7.4.1.4.10.4 A UPnP AV MediaServer control point must use a *RequestedCount* of 0 or 1 and *StartingIndex* of 0 when using CDS:Browse with the *BrowseMetadata* option.

M	L	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52] [55]

Improves expectations for CDS:Browse scenarios with *BrowseMetadata*.

7.4.1.4.10.5 A UPnP AV MediaServer device must always return one CDS object (as indicated in *TotalMatches* and *Result*) when successfully responding to a CDS:Browse request with the *BrowseMetadata* option, regardless of the *RequestedCount* value.

M	C	DMS	M-DMS	n/a	[52] [55]

7.4.1.4.10.6 If the UPnP AV MediaServer device returns more than zero CDS objects in a response to a CDS:Browse or CDS:Search query and if the UPnP AV MediaServer device does not provide an accurate value for the *TotalMatches* output argument, then the *TotalMatches* output value must be set to zero.

M	C	DMS	M-DMS	n/a	[52] [55]

This guideline allows control points to conclude that a *TotalMatches*==0 condition indicates that the UPnP AV MediaServer could not accurately calculate the value in cases where the UPnP AV MediaServer actually returned CDS objects.

Although some UPnP AV MediaServer implementations may choose to report the accurate *TotalMatches* value, at the expense of violating the 30-s timeout rule, DLNA does not recommend that implementation option. Subclause 7.3.10 DDC UPnP device responsiveness guideline indicates that a control point is allowed to terminate a SOAP response that exceeds a 30-s transmission time.

7.4.1.4.10.7 If a UPnP AV MediaServer device cannot find more than zero CDS objects (in 27 s, as described in 7.3.10.1), for a response to a CDS:Browse or CDS:Search query and if the UPnP AV MediaServer cannot calculate an accurate value for the *TotalMatches* output argument, then the UPnP AV MediaServer should return a SOAP error response code of 720 (cannot process the request).

S	C	DMS	M-DMS	n/a	[52] [55]

This guideline covers the scenario where a UPnP AV MediaServer can neither find any CDS objects that satisfy the query nor calculate the *TotalMatches* output argument accurately. Although some UPnP AV MediaServer implementations may choose to report the accurate *TotalMatches* value, at the expense of violating the 27 s timeout rule, such behaviour is not recommended for the same reason stated in the previous guideline.

7.4.1.4.10.8 A UPnP AV MediaServer control point should specify the desired number of CDS objects in the RequestedCount input argument of a CDS:Browse or CDS:Search query.

S	C	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52] [55]

This guideline recommends control points to request a reasonable number of CDS objects in a single CDS query. The number of CDS objects that can be displayed to the user at a single time is a good measure of reasonableness. Using a RequestedCount of zero may cause the transmission of a huge SOAP response, which is undesirable.

7.4.1.4.10.9 A UPnP AV MediaServer control point should specify smaller (about 10 to 30) RequestedCount input values for CDS:Browse and CDS:Search requests to receive a faster response time.

S	C	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52] [55]

Generally speaking, control points that specify smaller RequestedCount values will receive the response from the device sooner than if a larger value were specified.

7.4.1.4.10.10 A UPnP AV MediaServer control point must not assume that a UPnP MediaServer will return all of the CDS objects requested in a browse request.

M	R	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52] [55]

This places requirements on an AV MediaServer control point to not assume that its browse request will return all of the CDS objects it requested.

7.4.1.4.10.11 If a UPnP AV MediaServer control point wants to retrieve the remaining items in a reduced response, the UPnP AV MediaServer control point must issue additional browse requests to complete the original browse request for CDS objects.

M	R	DMP DMC +DN+ +UP+ +PR2+	M-DMP M-DMC M- DMU, M-DMD	MIU	[52] [55]

7.4.1.4.11 MM container update IDs event

UPnP AV MediaServer devices should implement behaviour for the CDS.ContainerUpdateIDs state variable.

S	L	DMS	M-DMS	n/a	[52]

This guideline is a benefit to both devices and control points, although lightweight UPnP AV MediaServer ContentDirectory Service (CDS) implementations may have difficulty implementing it. The rationale for this guideline stems from the fact that UPnP AV MediaServer control points can rely on the CDS.ContainerUpdateIDs state variable to minimize the number of CDS:Browse requests. A control point that relies solely on the CDS.SystemUpdateID state variable must browse the entire CDS hierarchy. Use of the

CDS.ContainerUpdateIDs state variable can limit the browse requests to the container objects that observed the metadata changes.

7.4.1.4.12 MM search capabilities

7.4.1.4.12.1 UPnP AV MediaServer entities that implement CDS:Search should support search queries with the following metadata properties:

- dc:title;
- dc:creator;
- upnp:class;
- res@protocolInfo;
- @refID.

S	L	DMS	M-DMS	n/a	[52]

This requirement describes the recommended search capabilities for a UPnP AV MediaServer that supports search operations.

7.4.1.4.12.2 All searchable properties must be listed in the return value of CDS:GetSearchCapabilities if the device implements CDS:Search.

M	R	DMS	M-DMS	n/a	[52]

This guideline mandates that supported search properties be discoverable.

7.4.1.4.12.3 If a UPnP AV MediaServer supports CDS:Search for an indicated property or a property with an indicated type, then it must minimally support the following operators for the specified property types. It should be note that rows in the table are not additive.

Table 15 – CDS – Search minimum support of operators

Property	Operators
@id	=, exists
@refID	=, exists
upnp:class	derivedfrom, exists
Any date, time, duration-based property types	<, <=, >=, >, =, !=, exists
All other string-based property types other than those listed in previous table entries	contains, =, exists
All URI-based property types	contains, =, exists
Integer or numerical property types	<, <=, >=, >, =, !=, exists
Boolean-based property types	=, !=, exists
All other attributes and elements	Exists

Vendors are free to apply additional CDS-normative operators for these properties or property types.

M	L	DMS	M-DMS	n/a	[52]

This guideline specifies the minimum behaviour and capabilities for various query operators.

Vendors cannot change (override) the default behaviour of these required operations as specified; but additions of standard operators are allowed.

Note that the exists operator only allows a value of "true" or "false" and those values are not quoted when used.

- VALID: @refID exists false
- INVALID: @refID exists 0
- INVALID: @refID exists "false"

7.4.1.4.12.4 If a UPnP AV MediaServer reports a searchable property for a particular data type, then a UPnP AV MediaServer must implement the associated operators for that data type.

M	L	DMS	M-DMS	n/a	[52]

For example, if a UPnP AV MediaServer reports its search capabilities to be only dc:title, then it only needs to implement the exists, contains, and = operators.

7.4.1.4.13 MM search all CDS objects for a media class

7.4.1.4.13.1 A UPnP AV MediaServer that implements the CDS:Search action should support the search with the following input parameters:

- ContainerID: 0;
- SearchCriteria: upnp:class derived from "[a upnp class for a supported media class]" and @refID exists false.

S	A	DMS	M-DMS	n/a	[52]

A ContentDirectory service may expose CDS objects in different collections, such as, lists by genres, lists by artists, lists of favourite items, etc. In such cases, reference items, which have @refID are used to represent CDS items that are references to other CDS items in the CDS hierarchy.

In some use case scenarios, a UPnP AV MediaServer control point wants to locate a set of CDS objects, excluding their duplicate references. For example, a UPnP AV MediaServer control point regenerates another presentation of CDS objects by using metadata properties of CDS objects. In this case, this search can be used.

It should be noted that live AV and audio tuner content are also located by searching for object.item.videoItem.videoBroadcast and object.item.audioItem.audioBroadcast (and their derived classes). It should be noted that the search results do not include the CDS container (i.e., ContainerID) that was specified in the request.

7.4.1.4.13.2 If the UPnP AV MediaServer supports a DLNA media class, it must support a corresponding object class as follows.

Table 16 – UPnP – Class for searching all CDS objects

Media class	upnp:class value
Audio	object.item.audioItem

Image	object.item.imageItem
AV	object.item.videoItem

The following is an example to search all video items which are stored in the ContentDirectory service.

- request: Search("0", "upnp:class derivedfrom "object.item.videoItem" and @refID exists false", "dc:date,upnp:genre,res@duration", 0, 40, "")
- response: Search("<DIDL-Lite xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/" xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/">
<item id="10" parentId="4" restricted="0">
<dc:title>Desire stones</dc:title>
<dc:date>2004-07-04T20:00:00</dc:date>
<upnp:genre>Movie</upnp:genre>

<upnp:class>object.item.videoItem</upnp:class>
<res protocolInfo="http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;DLNA.ORG_FLAGS=01100000000000000000000000000000" duration="0:59:53">http://192.168.1.1/res?id=10</res>
</item>
<item id="13" parentId="12" restricted="0">
<dc:title>Music Street in Asia</dc:title>
<dc:date>2004-07-04T23:30:00</dc:date>
<upnp:genre>Music</upnp:genre>

<upnp:class>object.item.videoItem</upnp:class>
<res protocolInfo="http-get:*.video/mpeg:DLNA.ORG_PN=MPEG_PS_NTSC;DLNA.ORG_FLAGS=01100000000000000000000000000000" duration="0:29:54">http://192.168.1.1/res?id=13</res>
</item>
</DIDL-Lite>", 2, 2, 10434)

M	A	DMS	M-DMS	n/a	n/a

7.4.1.4.13.3 UPnP AV MediaServer responses for CDS:Search should only list CDS containers once. Duplicate CDS containers should be omitted from the results set

Duplicate CDS containers are CDS containers that provide the same set of child CDS items.

S	A	DMS	M-DMS	n/a	[52]

Duplicate CDS containers may be implemented in a ContentDirectory service which exposes the same content in different groupings.

For example, a music library may expose the same set of music tracks in two different CDS containers. In this example, the CDS containers could have a hypothetical path from the root as follows.

- "0" => "All Albums" => "The Doors Greatest Hits"
- "0" => "All Artists" => "The Doors" => "The Doors Greatest Hits"

This guideline recommends the MediaServer to return either of these containers rather than both of them in the CDS:Search response where the SearchCriteria is "upnp:class derivedfrom "object.container.musicAlbum"".

7.4.1.4.13.4 The UPnP AV MediaServer that supports the recommended search criteria of guideline 7.4.1.4.13.1 must return a SearchCaps output value (from CDS:GetSearchCapabilities) that includes upnp:class and @refID.

Example:

- request: GetSearchCapabilities
- response: GetSearchCapabilities("upnp:class,@refID")

The ContentDirectory service of this MediaServer must provide the following properties and the values as metadata of the root Container.

- @searchable="1"
- <upnp:searchClass includeDerived="1">[upnp class]</upnp:searchClass> or
<upnp:searchClass includeDerived="0">[upnp class]</upnp:searchClass>

The following DIDL-Lite fragment is an example of metadata for a root Container in the ContentDirectory service that supports audio, image and AV media classes.

Example:

- <DIDL-Lite xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/" xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/">

<container id="0" parentID="-1" childCount="3" restricted="1"

searchable="1">

<dc:title>Root Container</dc:title>

<upnp:class>object.container</upnp:class>

<upnp:searchClass

includeDerived="1">object.item.audioItem</upnp:searchClass>

<upnp:searchClass

includeDerived="1">object.item.imageItem</upnp:searchClass>

<upnp:searchClass

includeDerived="1">object.item.videoItem</upnp:searchClass>

</container>

</DIDL-Lite>

M	R	DMS	M-DMS	n/a	[52]

A UPnP AV MediaServer control point can know whether a ContentDirectory service supports the search defined in guideline 7.4.1.4.13.1 by checking the search capabilities and the metadata of the root container.

7.4.1.4.14 MM keyword search templates

A UPnP AV MediaServer that implements the CDS:Search action should support the following types of CDS:Search requests, with the ContainerID input parameter set to "0":

- "dc:title contains " val1 " and @refID exists false";
- "dc:creator contains " val1 " and @refID exists false";
- "upnp:album contains " val1 " and @refID exists false".

S	A	DMS	M-DMS	n/a	[52]

Some control points with an alphanumeric input mechanism (keyboards, virtual keyboards, cell phone keypad, etc.) can benefit from keyword searching capabilities.

7.4.1.4.15 MM tuner container

7.4.1.4.15.1 A tuner should be represented as a container with a class of object.container or any derived class. A tuner container should have an associated name. The name is given by property dc:title.

S	A	DMS	M-DMS	n/a	[52]

It allows multiple tuners to be represented in CDS with a friendly name. Tuner channel ordering allows the control point to implement up/down by selecting the next item in the container.

See Annex B for recommendations on how to represent a tuner container.

Control points should note that in compliance with UPnP guidelines (7.4.1.4.9), conformant DMS devices will not return the <dlina:containerType> property unless it is specifically requested as part of a CDS:Browse filter argument.

7.4.1.4.15.2 A tuner container must have a property <dlina:containerType> and must have a value of Tuner_1_0.

The name space for DLNA defined properties must be "urn:schemas-dlna-org:metadata-1-0/" and the namespace prefix must be "dlna:".

The following is an example.

- <dlna:containerType xmlns:dlna="urn:schemas-dlna-org:metadata-1-0/">Tuner_1_0</dlna:containerType>

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.15.3 A tuner container must contain object items of class object.item.videoItem.videoBroadcast or object.item.audioItem.audioBroadcast or both (objects derived from either class also qualify).

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.15.4 The order of the object items (order of <item> elements in a CDS:Browse response) in a tuner container should correspond to the tuner up/down operation. For example, channel number or channel preset order.

S	A	DMS	M-DMS	n/a	[52]

7.4.1.4.16 MM audio tuner

If a UPnP AV MediaServer provides live audio content from a tuner, the UPnP AV MediaServer should use object.item.audioItem.audioBroadcast or any derived class as the upnp:class value.

S	A	DMS	M-DMS	n/a	[52]

These guidelines allow control points to identify content sourced from a tuner.

7.4.1.4.17 MM video tuner

If a UPnP AV MediaServer provides live video or audio/video content from a tuner, the UPnP AV MediaServer should use object.item.videoItem.videoBroadcast or any derived class as the upnp:class value.

S	A	DMS	M-DMS	n/a	[52]

See comment text for 0.

7.4.1.4.18 MM tuner properties – Channel title

The dc:title should describe the program content if available otherwise should contain the contents of the channelName CDS property. If channelName CDS property is not available the upnp:channelNr property should be used.

S	C	DMS	M-DMS	n/a	[52]

This is to clarify the meaning of title in context of a tuner. Some vendors may interpret title as channel name.

7.4.1.4.19 MM tuner properties – Channel number

7.4.1.4.19.1 The broadcast object item should have the associated property upnp:channelNr.

S	C	DMS	M-DMS	n/a	[52]

This guideline clarifies the intended usage of upnp:channelNr.

7.4.1.4.19.2 If upnp:channelNr is used, then each upnp:channelNr number must be unique within the context of its container.

M	C	DMS	M-DMS	n/a	[52]

7.4.1.4.20 MM tuner properties – Channel name

If the broadcast object item has the associated property upnp:channelName, then each upnp:channelName string should be unique within the context of its container. The channelName should be used to identify the channels, not the program content.

S	A	DMS	M-DMS	n/a	[52]

This guideline clarifies the intended usage of upnp:channelName.

7.4.1.4.21 MM tuner content URI

The channel selection and the connection to the tuner are invoked through the connection establishment to the URI of the resource associated with the broadcast object item.

M	A	DMS	M-DMS	n/a	[52]

This guideline essentially requires tuner content to be advertised and accessible through a URI.

7.4.1.4.22 MM tuner control point assumptions

The UPnP AV MediaServer control point should not assume that the currently viewed channel is the channel that it previously selected. Due to possible sharing of the tuner by multiple clients the channel may change without the client being aware of the change.

S	A	DMP DMC +DN+	M-DMP M-DMC M-DMD	MIU	[52]

Because there is no feed back mechanism between the tuner and the control point, it is not possible to know the current channel when there are multiple content receivers connected as clients or if the tuner is being used by a local output device.

7.4.1.4.23 MM TakeOut contents

7.4.1.4.23.1 A UPnP AV MediaServer may provide the DLNA defined property <dlina:takeOut> for a CDS object that belongs to a group of objects intended for unidirectional synchronization (from the server to a downloading endpoint).

The <dlina:takeOut > is a non-empty string value with a maximum length of 256 bytes.

The name space for DLNA defined properties must be "urn:schemas-dlna-org:metadata-1-0/" and the namespace prefix must be "dlina:".

O	A	DMS	M-DMS	n/a	[52]

The guidelines for takeOut enable unidirectional synchronization scenarios. This allows a device to easily find a group of content that is intended for download. Control points find groups by using CDS:Search and specifying the <dlina:takeOut> property with the desired group name.

The general assumption is that a control point (bundled with a download controller or an M-DMD) will find content with a particular group name and download it (unless it has already acquired it). The <dlina:takeOut> tag will generally persist on the DMS or M-DMS, so that if a CDS object is no longer part of a group, then the control point will know that it should remove the content from its local storage during the synchronization operation.

The process of assigning the value of the <dlina:takeOut> tag is vendor defined. One way is for the DMS or M-DMS to assign the value for a user when the user creates the group. Another way is to allow the user to name the group. An extremely simple way is to assign the same group name to all CDS objects.

DLNA suggests that DMS or M-DMS implementations provide a way for a user to specify a maximum storage quota when creating a group. This type of user interface feature is useful because users are generally aware that downloading endpoints have storage limitations for downloading groups.

It should be noted that this guideline does not imply any requirement that the downloader mirrors the CDS hierarchy when it downloads content from the CDS. This is especially true when the takeOut group includes CDS containers (i.e., the CDS container has the <dlina:takeOut> element.

7.4.1.4.23.2 A CDS object (advertised by a UPnP AV MediaServer) may contain more than one <dlina:takeOut> element.

O	A	DMS	M-DMS	n/a	[52]

A CDS object can belong to multiple groups.

This guideline provides for basic unidirectional synchronization of content from a DMS to UPnP control points. The primary mechanism used by a control point to identify content between reboots or application launches is through the object ID. The mechanism to indicate if content has been modified is to change the object ID but to change the value of the URI (i.e., value of the <res> element).

7.4.1.4.23.3 If a UPnP AV MediaServer provides the <dlina:takeOut> property to a CDS object and wants to indicate that the content is identical between reboots or application launch/shutdown, then it must not change the object ID of the CDS object.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.23.4 If a UPnP AV MediaServer control point is synchronizing a particular take-out group, then the control point must only transfer content binaries belonging to CDS objects that are part of the take-out group (i.e., If a UPnP AV MediaServer control point is not able to find a take-out CDS object that was synchronized on a previous session or if the CDS object of the same @id value no longer has the <dlina:takeOut> property with the same group name, then the control point must assume that the CDS object is no longer part of the take-out group.

M	A	+DN+	M-DMD	MIU	[52]

7.4.1.4.23.5 If a UPnP AV MediaServer control point is synchronizing a particular take-out group and finds a take-out CDS object that was not synchronized on a previous session, then it must attempt to transfer at least one content binary from the associated CDS object (i.e., if a UPnP AV MediaServer control point finds a take-out CDS object that was not synchronized on a previous session, then it must assume that the CDS object is a new member of the take-out group.)

M	A	+DN+	M-DMD	MIU	[52]

7.4.1.4.23.6 If a UPnP AV MediaServer provides the <dlina:takeOut> property to a CDS object and wants to indicate that a content binary is modified (for example, editing), it must change the @id value to a new and unique value, relative to the entire CDS hierarchy.

M	C	DMS	M-DMS	n/a	[52]

7.4.1.4.23.7 If a UPnP AV MediaServer supports the <dlina:takeOut> property, then it must support the DLNA-defined CDS:X_GetTakeOutGroupNames action.

The action's definition in the service description is defined below.

- <action>
- <name>X_GetTakeOutGroupNames</name>
- - <argumentList>
- - <argument>
- <name>GroupNames</name>
- <direction>out</direction>
- <relatedStateVariable>X_A_ARG_Type_GroupNames</relatedStateVariable>

- `</argument>`
- `</argumentList>`
- `</action>`

The X_A_ARG_TYPE_GroupNames state variable is defined below.

- `<stateVariable sendEvents="no">`
- `<name>X_A_ARG_Type_GroupNames</name>`
- `<dataType>string</dataType>`
- `</stateVariable>`

The GroupNames output argument is a comma-separated value list of all take-out group names.

M	A	DMS	M-DMS	n/a	[52]

The CDS:GetTakeOutGroupNames action allows a control point to easily acquire the list of group names for use in a CDS:Search request. DLNA has two normative search templates for finding groups. The only difference between the two is that vendors can specify a upnp:class value if desired.

7.4.1.4.23.8 If a UPnP AV MediaServer supports the `<dlna:takeOut>` property, then it must support searching the CDS as defined in 7.4.1.4.13 and the additional searches with the following input parameters:

- Additional Search #1 input parameters;
ContainerID = 0
SearchCriteria: @refID exists false and dlna:takeOut="[group name]"
- Additional Search #2 input parameters;
- ContainerID = 0;
- SearchCriteria: upnp:class derivedfrom "[a upnp class for a supported media class]" and @refID exists false and dlna:takeOut="[group name]".

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.23.9 A UPnP AV MediaServer that supports the search defined in guideline 7.4.1.4.23.8 must return the SearchCaps that includes the `<dlna:takeOut>` element to the CDS:GetSearchCapabilities action.

Example:

- request: CDS:GetSearchCapabilities()
- response: CDS:GetSearchCapabilities("upnp:class,@refID,dlna:takeOut")

M	R	DMS	M-DMS	n/a	[52]

A UPnP AV MediaServer control point can know whether a ContentDirectory service supports the search defined in guideline 7.4.1.4.23.8 by checking the search capabilities.

7.4.1.4.24 MM CDS containers and media collection binaries

7.4.1.4.24.1 Whenever possible a UPnP AV MediaServer should use a container object with a set of child item objects to indicate a media collection.

S	A	DMS	M-DMS	n/a	[52]

Using this model for media collections allows rendering devices to render media collections. Media collection file formats are useful in various ways, but v1.0 DMP's might lack the ability to parse media collection files.

7.4.1.4.24.2 A UPnP AV MediaServer may associate a <res> element with a CDS container or item such that the <res> element's URI value points to a media collection file conforming to a media format profile defined in the media collection profile guidelines given in Clause 11 of IEC 62481-2 (for example, DIDL_S, DIDL_V).

O	A	DMS	M-DMS	n/a	IEC 62481-2 [52]

Media collection files are a convenient way to allow a rendering device (for example, DMR) to play a media collection.

7.4.1.4.24.3 A UPnP AV MediaServer that advertises a <res> element for a media collection file should also use the res@dlna:trackTotal attribute, which is a ui4 value.

This guideline applies only to media collection files that are normative to the DLNA guidelines (for example, DIDL_S and DIDL_V).

S	A	DMS	M-DMS	n/a	[52]

Controlling devices (for example, DMC, M-DMC) can use this value when the calculating track index for the currently playing content in a DLNA PlayContainer URI operation.

7.4.1.4.24.4 If present, the value of res@dlna:trackTotal must equal the number of content entries (a.k.a sequenced tracks) in the media collection file.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.24.5 If a UPnP AV MediaServer wants to convey that the items (or tracks) in a media collection file have a 1:1 mapping to the child CDS objects in a particular container, then the UPnP AV MediaServer should associate the <res> element (for the media collection file) with the CDS container that has the child CDS objects. The ordering of the 1:1 mapping is the same for both the media collection file and the CDS objects in the CDS container (as obtained from a CDS.Browse request with an unspecified sorting criteria).

This methodology should be used even when the media collection file uses or references content intended for presentation-layer details. A UPnP AV MediaServer that uses this methodology claims an intent that the content in the media collection file maps to the child objects of the associated container.

It should be noted that CDS container in this guideline means any CDS object with the object.container or similarly derived upnp:class designation.

S	A	DMS	M-DMS	n/a	[52]

This guideline recommends that a UPnP AV MediaServer that wants to represent the child CDS objects encapsulated by a CDS container should associate a playlist file with the parent CDS container.

This guideline is only a recommendation because the file format of the media collection file could reference things intended for the presentation layer. For example, a slideshow file might have URI values that point to proprietary background music instructions and transitions. In such a case, the MediaServer associates the slideshow file with a container that was the parent of image items because the general intent is to have a 1:1 mapping.

7.4.1.4.24.6 If a UPnP AV MediaServer wants to convey that the items (or tracks) in a media collection file do not have a direct mapping to a particular container, then the MediaServer should associate the <res> element (for the media collection file) with a CDS item that has the object.item.playlistItem (or similarly derived upnp:class) designation.

This methodology should be used whenever the media collection file has an intentional mapping of content in multiple CDS containers. Likewise, this methodology should be used when the media collection file references content (that is not associated with presentation-layer details).

S	A	DMS	M-DMS	n/a	[52]

In cases where a vendor wants to provide a playlist file without implying any intent about the mapping of that content to a particular CDS container, then the vendor should associate the media collection file with a CDS item object. This particular model (while easier for the DMS or M-DMS) can disadvantage rendering devices (for example, DMP) that do not support media collection files because they rely solely on the CDS hierarchy to provide the rendering experience.

7.4.1.4.25 MM rendering media collection files

7.4.1.4.25.1 A rendering endpoint may support rendering of media collection files.

O	A	DMP DMR	M-DMP M-DMD	n/a	[54]

DMP, M-DMP, M-DMD, and DMR devices are not required to render media collection files.

7.4.1.4.25.2 A UPnP AV MediaRenderer should support playback of a DLNA PlayContainer URI.

S	A	DMR	n/a	n/a	[54]

DMR devices are not required to support rendering of a DLNA PlayContainer URI because the feature requires the DMR to have a MediaServer control point. However, this feature is recommended for a DMR because DLNA recommends that DMS devices expose media collections through CDS container objects that have a set of child CDS item objects.

Implementing this feature requires adherence to additional guidelines, such as those in 7.4.1.6.10 and 7.4.1.4.28.

7.4.1.4.25.3 If a UPnP AV MediaRenderer supports the DLNA PlayContainer URI operation, then it must specify the <dlna:X_DLNAcap> element (as a child of the <device> element that represents the MediaRenderer) with the playcontainer-depth-strict token.

More formally, the syntax of the capability ID is defined below.

Table 17 – Capability ID syntax

Capability ID	Description

M	A	DMS	M-DMS	n/a	[52]

DLNA PlaySingle URI values can only have a single protocolInfo value. Therefore, a single CDS object may have more than one <res> elements with the same DLNA PlaySingle URI value and the different protocolInfo values. This allows UPnP AV MediaServer control points to know the media formats that are available on the referenced CDS object.

7.4.1.4.26.3 A UPnP AV MediaServer may have more than one <res> element with the same DLNA PlaySingle URI associated with a CDS item object.

O	A	DMS	M-DMS	n/a	[52]

7.4.1.4.26.4 A CDS item that has one or more DLNA PlaySingle URI <res> elements may have other <res> elements that do not have a DLNA PlaySingle URI as a value.

O	A	DMS	M-DMS	n/a	[52]

A MediaServer's CDS item can mix <res> elements that have different types of URI values. For example, a virtual DMS may have DLNA PlaySingle URI values for original content served by a different DMS while using RTP and HTTP URI values for the converted content that the virtualizing DMS will serve.

7.4.1.4.26.5 A UPnP AV MediaServer must only associate DLNA PlaySingle URI <res> elements with CDS item objects. DLNA PlaySingle URI <res> elements must not be associated with CDS container objects. DLNA PlaySingle URI <res> elements must not reference CDS container objects.

M	A	DMS	M-DMS	n/a	[52]

DLNA PlaySingle URI values are allowed only when used in conjunction with audio, audio/video, or image content. Using DLNA PlaySingle URI <res> elements to reference CDS containers, media collection files, or other CDS objects with a DLNA PlaySingle URI <res> element can result in circular references.

7.4.1.4.26.6 The res@protocolInfo value of a DLNA PlaySingle URI <res> element must reference content that qualifies as audio, audio/video, or image content.

A res@protocolInfo value that identifies a media collection file or XHTML Print document is prohibited. Similarly, a <res> DLNA PlaySingle URI value that references a media collection file is also prohibited.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.26.7 A DLNA PlaySingle URI must only reference a CDS item object that does not have a DLNA PlaySingle URI <res> element.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.26.8 A DLNA PlaySingle URI must only reference a CDS item object with a upnp:class that is the same or derived from one of the following:

- object.item.audioItem;
- object.item.videoItem;
- object.item.imageItem.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.26.9 A CDS item that has a DLNA PlaySingle URI <res> element must have the same upnp:class value as the CDS object that is referenced or one of its super classes.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.26.10 A DLNA PlaySingle URI may be used with media format profiles that are not defined by DLNA.

O	A	DMS	M-DMS	n/a	[52]

The DLNA PlaySingle URI values can be used with undefined media formats. In such scenarios, the DLNA PlaySingle URI value still need to comply with all of the rules declared in 7.4.1.4.26.

7.4.1.4.26.11 The syntax of DLNA PlaySingle URI is as follows:

- dlna-playsingle-uri = "dlna-playsingle://" cds-udn "?" service-id-arg item-id-arg;
- cds-udn = <the UDN of the UPnP AV MediaServer device>;
- service-id-arg = "serviceId=" service-id-val;
- service-id-val = <service ID of the CDS belonging to the MediaServer>;
- item-id-arg = "&ItemID=" item-id-val;
- item-id-val = <The @id string value of the CDS item to be referenced. The CDS item must comply with guidelines 7.4.1.4.26.6 – 7.4.1.4.26.8>.

PlaySingleURI values must be case-sensitive values, even though the general URI syntax is a case-insensitive value. The URI must also be escaped as described by 7.4.1.3.10.

M	A	DMS	M-DMS	n/a	[52]

7.4.1.4.27 MM control point rules for DLNA PlaySingle URIs

7.4.1.4.27.1 A UPnP AV MediaRenderer control point must not invoke AVT:SetAVTransportURI with the *CurrentURI* input argument set to a DLNA PlaySingle URI.

M	A	DMC +PU+	M-DMC	MIU	[50] [54]

See 7.4.1.4.27.3 for more information.

7.4.1.4.27.2 DLNA device classes and capabilities with a UPnP AV MediaServer control point and appropriate transport layer components may support the rendering or transporting of content referenced by DLNA PlaySingle URI values.

O	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[55]

DMP, DMR, and Download Controllers are not required to render or transport DLNA PlaySingle URIs.

7.4.1.4.27.3 A UPnP AV MSCP+MRCP that wants to instruct a UPnP AV MediaRenderer to play a DLNA PlaySingle URI must resolve the DLNA PlaySingle URI to a URI for the actual audio, audio/video, or image content for use with the AVT:SetAVTransportURI request.

M	A	DMC	M-DMC	MIU	[50] [54] [55]

Prior to invoking a AVT:SetAVTransportURI, the control point must set up a UPnP AV connection between the DMR and the Content Source. Since a DLNA PlaySingle URI is a URI pointer to a CDS object (and is not a pointer to an actual content binary), the MediaRenderer control point does not have sufficient information to create the logical connection prior to invoking AVT:SetAVTransportURI.

7.4.1.4.28 MM DLNA PlayContainer URI

7.4.1.4.28.1 The syntax of a DLNA PlayContainer URI must be as described in the BNF notation. Assume no linear white space

- `dlna-playcontainer-uri` = "dlna-playcontainer.//" `cds-udn` "?" `service-id-arg` `container-id-arg` `first-item-id-arg` `first-item-index-arg` [`sort-arg`] [`max-depth-arg`]
- `cds-udn` = <the UDN of the UPnP AV MediaServer device>
- `service-id-arg` = "<sid>" `service-id-val`
- `service-id-val` = <service ID of the CDS belonging to the MediaServer>
- `container-id-arg` = "<cid>" `container-id-val`
- `container-id-val` = <The @id string value of the CDS container that represents the top container of the media collection.>
- `first-item-id-arg` = "<fid>" `first-item-id-val`
- `first-item-id-val` = <The @id string value of the first CDS item that will be played as part of the PlayContainer URI operation. Note that 7.4.1.4.25.3 (MM rendering media collection files) places restrictions on first-item-id-val. Specifically, if the strict-token (defined in 7.4.1.4.25.3) is "1", then first-item-id-val must be a CDS item that is an immediate child of container-id-val. If strict-token is "0", then first item-id-val must be a CDS item that is descended from container-id-val, whose parent container has a container depth that is less than or equal to the depth-token (defined in 7.4.1.4.25.3).>
- `first-item-index-arg` = "<fii>" `first-item-index-val` `first-item-index-val` = <a ui4 value, as used in a CDS:Browse request. This value must represent the zero-based index of the first-item-id-val CDS item, such that a CDS:Browse request where the following argument values will result in a response where the first returned CDS item has @id = first-item-id-val.
 - `ObjectID` = parent container of first-item-id-val
 - `BrowseFlag` = "BrowseDirectChildren",
 - `SortCriteria` = sort-arg, and
 - `StartingIndex` = first-item-index-val

- sort-arg = "&sc=" sort-val
- sort-val = <A *SortCriteria* string, as defined for a CDS:Browse request. Equivalently, the value must comply with the A_ARG_TYPE_SortCriteria syntax, defined by the ContentDirectory service. If sort-arg is omitted, then assume an effective value of an empty string.>
- max-depth-arg = "&md=" max-depth-val
- max-depth-val = <A ui4 value that specifies the maximum descent level in the container. If max-depth-arg is omitted, then the value infers an effective value of "0". Note that 7.4.1.4.25.3 (MM rendering media collection files) places restrictions on max-depth-arg. Specifically the max-depth-val must be less than or equal to the depth-token, as defined in 7.4.1.4.25.3>

The cds-udn, service-id-val, object-id-val, sort-val, object-index-val, and max-depth-val tokens must be URI-escaped according to [35].

Ordering of tokens is significant.

The maximum length of a DLNA PlayContainer URI is 1024 bytes.

M	A	DMC +PU+	M-DMC	MIU	[35] [52] [55]

This guideline defines the syntax of a DLNA PlayContainer URI.

7.4.1.4.28.2 If a UPnP AV MediaRenderer supports the DLNA PlayContainer URI operation, then the MediaRenderer must support the sort-arg and max-depth-arg portions of the dlna-playcontainer-uri syntax.

- sort-arg: This value determines the playback order, such that the playback order matches the order as observed from an equivalent CDS:Browse response.
- max-depth-arg: This value determines the maximum depth the MediaRenderer traverses when encountering child/descendent containers. A value of "0" means that child/descendent are not traversed.

M	C	DMR	n/a	n/a	[52] [55]

These guidelines clarify that control points are not required to specify all parameters in a DLNA PlayContainer URI, but a DMR must operate correctly if a control point does provide the optional parameters.

It should be noted that a DLNA PlayContainer URI does not necessarily mean that the DMR will issue a single type of CDS:Browse request. For example, the DMR's control point may progressively issue multiple CDS:Browse requests that result in fewer results to ensure that the 200 KB response limit is not exceeded by the DMS.

7.4.1.4.28.3 If a UPnP AV MediaRenderer supports the DLNA PlayContainer URI operation and the URI specifies a max-depth-val that is greater than depth-token, as defined in 7.4.1.4.25.3, then the MediaRenderer must return a UPnP AV error code of xxx.

M	A	DMR	n/a	n/a	[55]

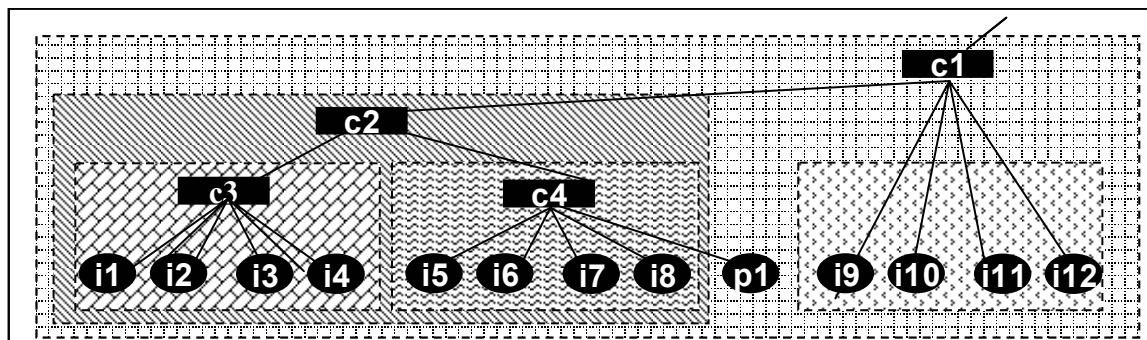
7.4.1.4.28.4 If performing a DLNA PlayContainer URI operation, a UPnP AV MediaRenderer must only play CDS items as part of the DLNA PlayContainer URI and must only render <res> elements of images, Audio-only, and AV content. This guideline works in conjunction with 7.4.1.3.31.1 (MM playcontainer-param (DLNA PlayContainer flag), which prohibits the

playcontainer-param from having a value of true for <res> elements of media collection binaries.

M	L	DMR	n/a	n/a	[54] [55]

7.4.1.4.28.5 MediaRenderer control points may omit the sort-arg, and max-depth-arg tokens in the dlina-playcontainer-uri syntax.

O	C	DMC +PU+	M-DMC	MIU	[54] [55]



Collection, start at	container-id-arg	first-item-id-arg	first-item-index-arg	max-depth-arg	Default Ordered tracks in collection	Track Index: item → AVT.CurrentTrack Index
 "i1"	"c3"	"i1"	"0"	Value > = "0" or omitted	i1, i2, i3, i4	i1→1, i2→2, i4→4
 "i6"	"c4"	"i6"	"1"	Value > = "0" or omitted	i6, i7, i8, i5 (*a)	i6→2, i8→4, i5→1
 "c3"	This is not permitted because first-item-id-arg shall be a CDS item. If first-item-id-arg was "i1", then playcontainer-strict shall be false (see next line/playlist)					
 "i1"	"c2"	"i1"	"0"	Value > = "1"	i1, i2, ..., i4, i5, ..., i8; (*a)	"i1"→1, "i8"→8
 "i12"	"c1"	"i12"	"4"; (*b)	Value = "0" or omitted	i12, i9, i10, i11	"i12"→4, "i9"→1, "i11"→3
 "i1"	"c1"	"i1"; (*c)	"0"	Value > = "2"	i1, i2, i3, i4, i5, i6, i7, i8, i9, i10, i11, i12	"i1"→1, "i4"→4, "i5"→5, "i8"→8, "i9"→10, "i12"→13; (*d)
 "i9"	"c1"	"i9"	"1"	Value > = "2"	i9, i10, i11, i12, i1, i2, i3, i4, i5, i6, i7, i8,	"i9"→10, "i12"→13, "i1"→1, "i4"→4, "i5"→5, "i8"→8; (*d)
(*a): Although "p1" is a child of "c4", it is a CDS object wit DIDL playlist file so it is automatically skipped (*b): "c2" is a child of "c1" that appears before "i9" when browsing the DMS (*c): This scenario is only permitted when playcontainer-strict is false (*d): Tracks after "i9" have an index number that is off by one because of p1, wich is a CDS object with a DIDL-Lite playlist file. It should also be noted that CDS containers are not rendered during a playcontainer URI operation						

IEC 1585/07

Figure 17 – Example of MM DLNA PlayContainer URI

7.4.1.4.29 MM control point rules for DLNA PlayContainer URI

7.4.1.4.29.1 A UPnP AV MSCP+MRCP may invoke AVT:SetAVTransportURI with the *CurrentURI* input argument set to a DLNA PlayContainer URI.

O	A	DMC	M-DMC	MIU	[50] [54]

7.4.1.4.29.2 If a UPnP AV MSCP+MRCP that invokes AVT:SetAVTransportURI with the *CurrentURI* input argument set to a DLNA PlayContainer URI, then the UPnP AV

MSCP+MRCP must only do so with UPnP AV MediaRenderer devices that have at least one protocolInfo value in the SinkProtocolInfo that has the playcontainer-param flag set to true.

SinkProtocolInfo refers specifically to the Sink output argument of CMS:GetProtocolInfo responses and the CMS.SinkProtocolInfo state variable.

M	A	DMC	M-DMC	MIU	[50] [51] [54]

See 7.4.1.3.23.2 and 7.4.1.3.31 for more information.

7.4.1.5 Basic connection management (BCM) guidelines

7.4.1.5.1 MM/BCM UPnP AV connection rules

7.4.1.5.1.1 UPnP AV MediaServers may provide support for basic connection management (BCM).

O	A	DMS	M-DMS	n/a	[51] [55]

DMS devices are not required to support basic connection management (BCM). A DMS with BCM reports the existence of transport layer connections by using UPnP AV connections, which may be enumerated and terminated. This feature helps third-party controllers and DMPs offer the users the ability to manage the connections according to their own desires (mainly for termination). This feature does not require any control point to support basic connection management in order to operate basic browse/play operations. This feature does not require a control point to interact with the user.

Transport clients are encouraged to implement these guidelines to allow MediaServers with BCM support to properly deliver BCM functionality.

7.5.5.2.36 MT/BCM HTTP Header:peerManager.dlna.org

7.5.5.2.37 MT/BCM HTTP Header:friendlyName dlna.org

7.5.8.2.26 MT RTP/BCM RTSP peerManager.dlna.org

7.5.8.2.27 MT RTP/BCM RTSP friendlyName dlna.org

The DLNA guidelines use the AVT:Stop action to terminate streams on a DMR instead of using CMS:ConnectionComplete.

7.4.1.5.1.2 If a UPnP AV MediaServer supports the BCM, then it must create a UPnP AV connection with a unique Connection ID for each new transport layer request corresponding to a content request, including background, interactive, and streaming requests (creating UPnP AV connections for upload-related content transfer process is outside the scope).

The list of transport layer requests corresponding to a media stream request includes HTTP GET requests and the RTSP SETUP command. The list of transport layer requests excludes HTTP HEAD and HTTP POST requests and all HTTP requests not related to a content URI (i.e. URI for a <res> element exposed by a UPnP AV MediaServer).

M	A	DMS	M-DMS	n/a	[51] [55]

The creation of UPnP AV connection is a result of a transport layer (HTTP or RTSP) connection request for a new media stream (see 7.4.1.5.2 for request for an existing media stream). All other non-media stream connection requests do not require the creation of UPnP AV connection. The media stream is identified by the content URI in the transport layer request.

CMS:PrepareForConnection is not required for the creation of a UPnP AV connection. However, using CMS:PrepareForConnection in conjunction with the creation of a transport layer is a permissible way to create UPnP AV Connection. It should be noted that a UPnP AV MediaServer control point is not required to call the optional CMS:PrepareForConnection. Hence, a DMS with BCM that also implements CMS:PrepareForConnection must also be able to create the UPnP AV connection without requiring a control point to call CMS:PrepareForConnection execution.

The UPnP AV media server should internally store all necessary information along with the newly created Connection ID for later actions (CMS:ConnectionComplete and CMS:GetCurrentConnectionInfo). For example, relevant informations are, but not limited to, protocolInfo, URI, and TCP/IP socket handles. The protocolInfo is already available in the DMS to satisfy 0 MT HTTP Header: contentFeatures.dlna.org.

7.4.1.5.1.3 A UPnP AV MediaServer that assigns a connection ID value in the range from "1" to "2147483647" should assign the value in an increasing manner with a rollover value of "1" when the vendor-defined-maximum-value is exceeded.

The vendor-defined-maximum-value is a vendor-defined value that is greater than, or equal to, "65535" and less than, or equal to, "2147483647".

S	A	DMS	M-DMS	n/a	[51] [55]

7.4.1.5.1.4 A UPnP AV MediaServer that assigns a connection ID value in the range from "1" to "2147483647" should start value assignment with a random value.

S	A	DMS	M-DMS	n/a	[51] [55]

7.4.1.5.1.5 If a UPnP AV MediaServer supports BCM, then it must expose the UPnP AV connections using CMS:GetCurrentConnectionIDs.

M	A	DMS	M-DMS	n/a	[51] [55]

7.4.1.5.1.6 If a UPnP AV MediaServer supports BCM, then it must remove the connection ID from the list of connections provided in CMS:GetCurrentConnectionIDs, when all transport layer connections (associated with the connection ID) are closed/terminated.

M	A	DMS	M-DMS	n/a	[51] [55]

The guideline clarifies the life-cycle of a connection ID. An active and advertised connection ID must be associated with at least one active media transport connection.

7.4.1.5.1.7 If a UPnP AV MediaServer supports BCM, then it must maintain the CMS current connectionIDs state variable and deliver an event for the ConnectionManager service to signal that an update to the connection information has occurred.

M	R	DMS	M-DMS	n/a	[51] [55]

The guideline repeats the UPnP AV requirements.

7.4.1.5.1.8 If a UPnP AV MediaServer supports BCM, then it must implement CMS:GetCurrentConnectionInfo for all UPnP AV Connections reported by CMS:GetCurrentConnectionIDs.

The following arguments and values are returned by CMS:GetCurrentConnectionInfo.

- RemoteProtocolInfo contains the res@protocolInfo of the transported content on the given connection. The value corresponds to the protocolInfo used to set the contentFeatures.dlna.org HTTP header or RSTP header according to 7.5.5.2.10 and 7.5.8.2.65;
- PeerConnectionManager contains
 - an empty string if the information is not available; or
 - the PeerConnectionManager value provided by a DMR using the peerManager.dlna.org HTTP header (defined in 7.5.5.2.36) or the peerManager.dlna.org RTSP header (defined in 7.5.8.2.26); or
 - the friendly name value provided by a DMP or M-DMP (that implements this version or a newer version of the DLNA interoperability guidelines) using the friendlyName.dlna.org HTTP header (defined in 7.5.5.2.37) or the friendlyName.dlna.org RTSP header (defined in 7.5.8.2.27);
- PeerConnectionID is set to "-1";
- Direction is set to "OUT".

M	A	DMS	M-DMS	n/a	[51] [55]

This guideline specifies the values returned by CMS:GetCurrentConnectionInfo for a given connection ID. Most arguments are compliant with the UPnP AV Architecture with the exception of PeerConnectionManager.

CMS:PrepareForConnection may also provide the PeerConnectionManager value when executed previously for the specified UPnP AV connection.

7.4.1.5.1.9 If a UPnP AV MediaServer supports BCM, then it must use the following notation to advertise a DMP or M-DMP user friendly name (when provided) in the PeerConnectionManager parameter, returned by CMS:GetCurrentConnectionInfo:

- peerconnectionmanager_arg = "fnam" ":" friendly-name "/";
- friendly-name = <string, limited to 128 bytes in its UTF-8 encoded form>.

M	A	DMS	M-DMS	n/a	[51] [55]

The DMP user friendly name is provided on the media transport layer connection (see 7.5.5.2.37 or 7.5.8.2.27) and is used to identify the content receiver connected to the DMS or M-DMS.

The format of the PeerConnectionManager is different from the UPnP Architecture in such a way that it cannot be confused with a non-existing CMS Service (i.e., no CMS on DMP).

Example: "fnam:LivingRoom TV/" where "fnam" replaces "uuid" and the ServiceId token is left empty.

7.4.1.5.1.10 If a UPnP AV MediaServer supports BCM, then it must implement CMS:ConnectionComplete.

M	C	DMS	M-DMS	n/a	[51] [55]

The requirement for CMS:ConnectionComplete does not imply to support and implement CMS:PrepareForConnection in the ConnectionManager service.

CMS:ConnectionComplete is the UPnP action that instructs a UPnP MediaServer to tear down a UPnP AV Connection (and its underlying transport layer connection) when it is no longer required. When CMS:ConnectionComplete is executed for a given Connection ID, the device should respond with a success and terminate the associated HTTP or RTSP transport connections related to the UPnP AV connection. Subsequently, all resources related to the connection are released. This behaviour is similar to a termination initiated by the connection creator on the content receiver endpoint. When a connection is terminated (for example, DMP performs a stop media operation, AVT:Stop is invoked on a DMR, the transport layer connection is broken, CMS:ConnectionComplete is called, etc.), it is expected that proper resource clean-up takes place.

7.4.1.5.2 MM/BCM UPnP AV connection rules for HTTP

If a UPnP AV MediaServer supports BCM, then it must not create a new UPnP AV connection when the ConnectionID is provided in the HTTP request using the scid.dlna.org header. Instead it must re-use the provided ConnectionID.

M	L	DMS	M-DMS	n/a	n/a

This guideline provides a mechanism to identify multiple transport layer request separated in time but related to the same media stream (identified by the URI). For example, during a pause operation and/or consecutive random access requests the HTTP client may indicate in the HTTP get request that all transport layer requests are related to the same media stream which does not require a new ConnectionID. With such mechanism the UPnP AV MediaServer is not required to maintain a local table containing the most recent requested URI with the associated ConnectionID per HTTP client. It is the HTTP client's responsibility to indicate the relationship between HTTP requests.

7.4.1.5.3 MM/BCM UPnP AV connection rules for RTP

If a UPnP AV MediaServer supports BCM and it supports RTP media transport, then it must not create a new UPnP AV Connection when the ConnectionID is provided in the RTSP session-id header. Instead it must re-use the provided ConnectionID.

M	L	DMS	M-DMS	n/a	n/a

7.4.1.5.4 MM/BCM-DMS CMS:ConnectionComplete and closing transport connections

7.4.1.5.4.1 If a UPnP AV MediaServer supports BCM, then when it responds successfully to a CMS:ConnectionComplete request for a valid Connection ID it must close all related transport layer connections.

Likewise, the corresponding connection ID must be removed from the list of connections provided in CMS:GetCurrentConnectionIDs.

M	A	DMS	M-DMS	n/a	n/a

This guideline clarifies the expected behaviour when UPnP MediaServers response successfully to a CMS:ConnectionComplete. The guideline does not prohibit a UPnP MediaServer to return an error code (for example, 704 local restriction) to indicate the impossibility to close the transport layer connections.

7.4.1.5.4.2 If a UPnP AV MediaServer supports BCM, then it must return 704 local restriction to the CMS:ConnectionComplete request if it does not accept to close all related transport layer connections.

M	A	DMS	M-DMS	n/a	n/a

7.4.1.6 MediaRenderer device requirements

7.4.1.6.1 MM UPnP AV MediaRenderer CMS:GetProtocolInfo behaviour

7.4.1.6.1.1 A UPnP AV MediaRenderer that lists a protocolInfo in its SinkProtocolInfo must be able to render that type of content when given a URI for that type of content.

SinkProtocolInfo refers specifically to the sink output argument of CMS:GetProtocolInfo responses and the CMS.SinkProtocolInfo state variable.

M	C	DMR	n/a	n/a	[51] [54]

DMR devices that claim support for a DLNA media format profile must be able to render that content when instructed to do so by a DMC. A DMR device that supports playback of content but requires a control point to send a URI to a media collection file or a DLNA PlayContainer URI in the AVT:SetAVTransportURI request is not permissible.

7.4.1.6.1.2 A UPnP AV MediaRenderer that supports playback of media collection files must include the media collection file's protocolInfo value in its list of SinkProtocolInfo (as observed from the sink output argument in CMS:GetProtocolInfo responses and its CMS.SinkProtocolInfo state variable).

M	A	DMR	n/a	n/a	[51] [54]

DMR uses protocolInfo values to report support for media collection files. Media collection files have mime types and media format profile IDs. DMR devices are not required to support playback of media collection files because the feature requires a DMR to implement a MediaServer control point.

7.4.1.6.2 MM AVTransport default instance

7.4.1.6.2.1 A UPnP AV MediaRenderer device must support a virtual instance of the AVTransport service that has an InstanceID equal to 0.

M	R	DMR	n/a	n/a	[50]

7.4.1.6.2.2 A UPnP AV MediaRenderer control point may use the InstanceID with a value of 0 to control the AVTransport service.

O	R	DMC +PU+	M-DMC	MIU	[50]

There is no need to invoke CMS:PreparForConnection when using default instances.

7.4.1.6.3 MM RenderingControl default instance

7.4.1.6.3.1 A UPnP AV MediaRenderer device must support a virtual instance of the RenderingControl service that has an InstanceID equal to 0.

M	R	DMR	n/a	n/a	[60]

The "default" instance (InstanceID=0) of the RenderingControl service is used to control the rendering characteristics of the "post-mixed" stream (i.e., all input content as a whole).

7.4.1.6.3.2 A UPnP AV MediaRenderer control point may use the InstanceID with a value of 0 to control the RenderingControl service.

O	R	DMC +PU+	M-DMC	M/U	[60]

There is no need to invoke CMS:PrepareForConnection when using default instances.

7.4.1.6.4 MM AVTransport multiple instances

A UPnP AV MediaRenderer device may support multiple virtual instances of the AVTransport service.

O	R	DMR	n/a	n/a	[51]

Multiple virtual instances of the AVTransport service must include the default instance (InstanceID=0) according to 7.4.1.6.2.1. Multiple virtual instances of the AVTransport service are permitted but DLNA defines no interoperability guidelines for InstanceIDs that are not equal to "0".

7.4.1.6.5 MM RenderingControl multiple instances

A UPnP AV MediaRenderer device may support multiple virtual instances of the RenderingControl service.

O	R	DMR	n/a	n/a	[51]

Multiple virtual instances of the RenderingControl service must include the default instance (ID=0) according to 7.4.1.6.3.1. Multiple virtual instances of the RenderingControl service are permitted but DLNA defines no interoperability guidelines for InstanceIDs that are not equal to '0'.

7.4.1.6.6 MM LastChange

UPnP MediaRenderers must properly implement the AVT.LastChange and RCS.LastChange evented state variables.

M	R	DMR	n/a	n/a	[50] [60]

This guideline enables control points the ability to rely on UPnP events and/or UPnP actions to track the state changes of the MediaRenderer device.

7.4.1.6.7 MM LastChange frequency

UPnP MediaRenderers should send no more than one AVT.LastChange or RCS.LastChange event for a single AVTransport or RenderingControl service action.

S	A	DMR	n/a	n/a	[50] [60]

Although UPnP AV has a maximum frequency for sending AVTransport and RenderingControl events, the preference is to send fewer events to reduce the load on control points.

7.4.1.6.8 MM AVT – SetAVTransportURI action availability

If a UPnP AV MediaRenderer receives an AVT:SetAVTransportURI request when the AVT.TransportState instance state variable is not "STOPPED" or "NO_MEDIA_PRESENT", the MediaRenderer must return an error 705 (transport is locked).

M	L	DMR	n/a	n/a	[50] [54]

If a MediaRenderer control point encounters the 705 error code, then it may invoke AVT:Stop to stop the transport layer and then retry the AVT:SetAVTransportURI request.

There may be reasons for the MediaRenderer to return an error, even when the transport state is STOPPED.

7.4.1.6.9 MM MediaRenderer selects a different <res> element

7.4.1.6.9.1 A UPnP AV MediaRenderer that receives an AVT:SetAVTransportURI request may override the CurrentURI input argument by selecting one of the <res> elements specified in the CurrentURIMetaData input argument.

O	A	DMR	n/a	n/a	[50] [54]

The general purpose is to allow a DMC to recommend a URI using the CurrentURI argument while providing alternate choices to the DMR through the CurrentURIMetaData of the AVT:SetAVTransportURI action.

If a DMS exposes multiple <res> elements, the DMC may not be able to determine the preferred <res> element for a given DMS / DMR pair. Examples of multiple <res> elements include

- HTTP and RTP media transports;
- transcoded content;
- multi-homed content;
- PS and ES versions of content (RTP media transport only);
- transrated content available at different bitrates.

CMS:GetProtocolInfo alone does not provide sufficient information for the DMC to always select the most appropriate URI to pass to AVT:AVTransportSetURI.

7.4.1.6.9.2 If a MediaRenderer control point calls AVT:SetAVTransportURI and provides a DIDL-Lite XML fragment for the CurrentURIMetaData argument, then the DIDL-Lite fragment should include all of the <res> elements associated with the CDS object.

S	L	DMC +PU+	M-DMC	MIU	[50] [54]

The DIDL-Lite XML fragment will at least have the <res> element for the URI specified in request, as required by 7.4.1.6.14.10. Additional <res> elements are recommended to allow the MediaRenderer to choose a different URI for playback.

7.4.1.6.10 MM MediaRenderer and DLNA PlayContainer URI

7.4.1.6.10.1 A UPnP AV MediaRenderer that supports playback of a DLNA PlayContainer URI must use a playcontainer-param set to true in the 4th field protocolInfo parameter of individual protocolInfo to indicate that the UPnP AV MediaRenderer will render that type of content when given a DLNA PlayContainer URI. This guideline specifically applies to the protocolInfo values listed in the Sink output argument of CMS:GetProtocolInfo responses and the CMS.SinkProtocolInfo state variable.

All guidelines in 7.4.1.6.10 apply in the context of a UPnP AV MediaRenderer supports the rendering of DLNA PlayContainer URIs.

All guidelines in 7.4.1.6.10 apply in conjunction with all other guidelines that govern a UPnP AV MediaServer control point.

M	A	DMR	n/a	n/a	[51] [54]

A DMR indicates support for DLNA PlayContainer URI values by using the playcontainer-param flag. ProtocolInfo values that omit this parameter indicate that the DMR will not play that type of content as part of a DLNA PlayContainer URI operation.

Setting the playcontainer-param flag to true does not mean that a DMR requires the media type to be played only through a DLNA PlayContainer URI operation (i.e., DMC can specify a URI that points to content of the specified protocolInfo or DMC can specify a DLNA PlayContainer URI that will result in the playback of content with the specified protocolInfo).

7.4.1.6.10.2 If a UPnP AV MediaRenderer supports playback of a DLNA PlayContainer URI, then it must support playback of a DLNA PlaySingle URI as part of the DLNA PlayContainer URI operation.

M	A	DMR	n/a	n/a	[54]

A DMR that can play a DLNA PlayContainer URI already has a MediaServer control point that enables it to browse a DMS.

7.4.1.6.10.3 A UPnP AV MediaRenderer that supports rendering of DLNA PlayContainer URIs must support rendering of a CDS object, if the CDS object meets the following criteria:

- the CDS object falls into the media collection that is defined by the DLNA PlayContainer URI;
- the CDS object has a <res> element has a res@protocolInfo that indicates a supported media format profile, as described by 7.4.1.6.1.

M	A	DMR	n/a	n/a	[52] [54]

This guideline essentially says that a DMR must render content that is supported by the DMR and can be found by traversing the CDS in the manner prescribed for a DLNA PlayContainer URI.

7.4.1.6.11 MM DMR AVT state variables and UPnP AV connections

7.4.1.6.11.1 A UPnP AV MediaRenderer that renders an audio, image, or audio/video content binary (regardless of whether it is part of a media collection file or a DLNA PlayContainer URI or specified directly by a MediaRenderer control point), must do the following:

- the AVT.AVTransportURI value must be the URI provided in the *CurrentURI* input argument of the AVT:SetAVTransportURI request or must be a URI obtained from a <res> element in the *CurrentURIMetaData* of the same request;

- the UPnP AV connection's information (i.e., *protocolInfo* and other information of *ConnectionID* "0") must correspond to the URI in the *AVT.CurrentTrackURI* virtual instance state variable.

M	C	DMR	n/a	n/a	[50] [54]

This allows control points to easily acquire the *protocolInfo* for currently rendered content. Note that media collection files (such as *DIDL_S* and *DIDL_V*) do not qualify as audio, image, or audio/video content binaries.

7.4.1.6.9.1 permits a DMR to override the *CurrentURI* with a URI from the *CurrentURIMetaData* argument.

7.4.1.6.11.2 A UPnP AV MediaRenderer that renders a content binary that is not part of a DLNA PlayContainer URI or media collection file must use the following mapping for virtual instance state variables and UPnP AV connection:

- the *AVT.CurrentTrackURI*: value must be equal to the *AVT.AVTransportURI* value;
- the *AVT.CurrentTrack* and *AVT.NumberOfTracks* must be "1"

M	C	DMR	n/a	n/a	[50] [54]

In conjunction with other DLNA guidelines that define general rules for ensuring consistency between different *AVTransport* state variables, this guideline clarifies how those other guidelines apply specifically, in the case where a control point instructs a DMR to render a single content binary.

7.4.1.6.11.3 A UPnP AV MediaRenderer that renders a media collection file uses the following mapping for virtual instance state variables and UPnP AV connection.

- The *AVT.CurrentTrackURI* value must be the URI for the content binary that is currently being rendered. (i.e., the URI for the actual audio-only, audio/video, or image content that is actually rendered).
- The *AVT.CurrentTrack* value must be the index within the media collection file, where the URI for the content entry can be found. This value is 1-based.
- The *AVT.NumberOfTracks* value must be the number of content entries in the media collection file (which is often progressively calculated).

M	C	DMR	n/a	n/a	[50] [54]

In conjunction with other DLNA guidelines that define general rules for ensuring consistency between different *AVTransport* state variables, this guideline clarifies how those other guidelines apply specifically, in the case of media collection files.

It should be noted that the *AVT.NumberOfTracks* virtual instance state variable is often updated progressively because it is often difficult to count the number of tracks for all types of media collection files.

7.4.1.6.11.4 A UPnP AV MediaRenderer that renders a DLNA PlayContainer URI operation uses the following mapping for virtual instance state variables and UPnP AV connection.

- The *AVT.CurrentTrackURI*: must be the URI for the content binary that is currently being rendered. (i.e., the URI for the actual audio-only, audio/video, or image content that is being rendered).
- The *AVT.CurrentTrack* must be the *PlayContainerTrackIndex* of the CDS object that is being rendered.
- The *AVT.NumberOfTracks* must be the *PlayContainerTotalTracks* (which is often progressively calculated).

PlayContainerTrackIndex is defined as the index of a rendered content, relative to the sequenced set of content, represented by the DLNA PlayContainer URI.

PlayContainerTrackTotal is defined as the last index of rendered content in the sequenced set of content, represented by the DLNA PlayContainer URI.

PlayContainerTrackIndex and PlayContainerTotalTracks must be calculated in a manner consistent with these rules.

- Index calculation must always be done with a preorder traversal of the CDS hierarchy, such that the first CDS object that is played must have an index of "1". A preorder traversal means that for a given CDS:Browse request (with sort-args applied), left subtrees and leaves (i.e. CDS objects that appear earlier in the CDS:Browse response) are processed before right subtrees and leaves (i.e., CDS objects that appear later in the CDS:Browse response).
- If encountered during traversal, CDS items must always have a count of "1"; regardless of whether they are played or skipped. A skipped CDS item is one where none of the CDS item's <res> elements are rendered by the MediaRenderer. If the CDS item has zero <res> elements, it must be considered skipped.
- A CDS item with a media collection binary must always have a count of "1". Furthermore these CDS items must be skipped. (See 0 MM DLNA PlayContainer URI for information on the prohibition of playing media collection binaries.)
- If rendered, a CDS item must automatically render at least one <res> elements associated with the CDS item. If multiple <res> elements are rendered, then they must be rendered simultaneously (for example, a MediaRenderer does not render the same content (in different profiles or transports) multiple times, but a MediaRenderer is permitted to render an image in conjunction with an audio at the same time).
- If encountered during traversal, a CDS container must have a count of "0" because <res> elements of a CDS container must never be played. (See 0 MM DLNA PlayContainer URI for information on the restriction for playing only CDS items.)

A CDS object must be counted if and only if there are max-depth-val or fewer CDS containers between the CDS object and the CDS container identified by container-id-val, as defined in 7.4.1.4.28. (i.e., these are the CDS objects that are traversed.)

M	C	DMR	n/a	n/a	[50] [54]

In conjunction with other DLNA guidelines that define general rules for ensuring consistency between different AVTransport state variables, this guideline clarifies how those other guidelines apply specifically, in the case of DLNA PlayContainer URI values.

Note that the AVT.NumberOfTracks virtual instance state variable is often updated progressively because it is often difficult to count the number of tracks for all types of media collections, whether they be represented through a DLNA PlayContainer URI or a media collection file.

7.4.1.6.12 MM GetMediaInfo behaviour

7.4.1.6.12.1 The CurrentURI output argument of the AVT:GetMediaInfo action must return one of the following.

- The URI provided in the CurrentURI input argument of the AVT:SetAVTransportURI request.
- The URI obtained from a <res> element in the CurrentURIMetaData argument of the AVT:SetAVTransportURI request.

Likewise, the value of the AVT.AVTransportURI instance state variable must have the same URI value as the CurrentURI output argument of AVT:GetMediaInfo response.

M	R	DMR	n/a	n/a	[50]

This requirement repeats what is already stated in the AVTransport specification. Some have accidentally mistaken the CurrentURI output argument and the AVT.AVTransportURI values to be always equivalent to the TrackURI output argument of AVT:GetPositionInfo and the AVT.CurrentTrackURI instance state variable.

7.4.1.6.9.1 permits a DMR to override the CurrentURI with a URI from the CurrentURIMetaData argument.

7.4.1.6.12.2 The NrTracks output argument of the AVT:GetMediaInfo action must return the same value as the AVT.NumberOfTracks instance state variable.

M	R	DMR	n/a	n/a	[50]

These guidelines specify the behaviour for reporting the value for NrTracks (and AVT.NumberOfTracks).

7.4.1.6.12.3 If the AVT.AVTransportURI instance state variable has an empty value, then the NrTracks output argument and the AVT.NumberOfTracks instance state variable must be "0".

M	R	DMR	n/a	n/a	[50]

7.4.1.6.12.4 If the AVT.AVTransportURI instance state variable point to content and that content is neither a media collection file nor a DLNA PlayContainer URI, then the *NrTracks* output argument and the AVT.NumberOfTracks instance state variable must be "1".

M	R	DMR	n/a	n/a	[50]

This guideline works in conjunction with 7.4.1.6.9.1 and 7.4.1.6.11.2.

7.4.1.6.12.5 If the AVT.AVTransportURI instance state variable points to a media collection, then the NrTracks output argument and the AVT.NumberOfTracks instance state variable should be the number of known tracks in the media collection.

MediaRenderers often know the number of tracks in a media collection through counting the playback items or by using some metadata within the media collection that provides the value immediately.

S	R	DMR	n/a	n/a	[50]

This guideline strongly recommends that a MediaRenderer provide the number of playback items in the current media collection. Sometimes the final value is acquired in an immediate fashion; other times, the value has to be acquired through a counting process. The manner in which the MediaRenderer acquires the total track count is up to the implementer, but the obligation to provide an accurate value exists according to the intentions of the AVTransport specification.

See 7.4.1.6.11.4 regarding the algorithm for calculating track total and track index values.

It should be noted that reporting an empty string is not allowed because an empty string is not a valid value for the ui4 data type.

7.4.1.6.12.6 In conjunction with 7.4.1.6.12.5, a MediaRenderer may progressively update the values of NrTracks and AVT.NumberOfTracks, such that the total count is not provided immediately.

The definition of progressively update is that the MediaRenderer updates the state variables such that a control point is able to receive GENA events at a rate that does not exceed one GENA event every 0,2 s and the control point is able to acquire the values using AVT:GetPositionInfo.

O	R	DMR	n/a	n/a	[50]

Some MediaRenderers may not be able to immediately provide a count of the playback items in a media collection. The AVTransport specification takes this into account and permits MediaRenderers to update the value in a progressive manner.

7.4.1.6.12.7 In conjunction with 7.4.1.6.12.6, a MediaRenderer that progressively updates the values of NrTracks and AVT.NumberOfTracks should provide the total count within 10 s of the previous call to AVT:SetAVTransportURI.

S	L	DMR	n/a	n/a	[50]

DLNA recommends that MediaRenderers provide the count in a timely manner. Control points often display such information to a user and sometimes use the track count information to determine what a user can or cannot do.

This guideline is not required because media collection files can be very large and CDS hierarchies can be very complex.

7.4.1.6.12.8 The MediaDuration output argument of the AVT:GetMediaInfo action must match the value specified in the AVT.CurrentMediaDuration instance state variable.

M	P	DMR	n/a	n/a	[50]

These guidelines specify the expectations for reporting the media duration. Sometimes the final value is acquired in an immediate fashion; other times, the value has to be acquired through a summation process.

Note that this guideline places no mandatory requirement on the accuracy of the total media duration. If the MediaRenderer is not able to provide a value, then the "NOT_IMPLEMENTED" value is used.

7.4.1.6.12.9 The MediaDuration output argument and the AVT.CurrentMediaDuration instance state variable should have a value that conforms to one of the following:

- represents the total duration for the content specified in AVT.AVTransportURI;
- if the content is a media collection, the total duration is the sum of all known playback durations for each item in the media collection;
- has a value of "NOT_IMPLEMENTED".

S	R	DMR	n/a	n/a	[50]

MediaRenderers often know the playback duration, of items in a media collection, through summing the durations progressively or by using some metadata within the media collection that provides the value in an immediate fashion.

7.4.1.6.12.10 In conjunction with 7.4.1.6.12.9, a MediaRenderer may progressively update the values of MediaDuration and AVT.CurrentMediaDuration, such that the total sum is not provided immediately.

The definition of progressively update is that the MediaRenderer updates the state variable (and appropriately sends GENA events) at a rate that does not exceed one GENA event every 0,2 s.

O	R	DMR	n/a	n/a	[50]

Some MediaRenderers may not be able to immediately provide the duration sum for all playback items in a media collection. The AVTransport specification takes this into account and permits MediaRenderers to update the value in a progressive manner.

7.4.1.6.12.11 In conjunction with 7.4.1.6.12.10, a MediaRenderer that progressively updates the values of MediaDuration and AVT.CurrentMediaDuration should provide the total sum within 10 s of the previous call to AVT:SetAVTransportURI.

S	L	DMR	n/a	n/a	[50]

DLNA recommends that MediaRenderers provide the duration sum in a timely manner.

7.4.1.6.12.12 Information returned by AVT:GetMediaInfo should be as accurate as possible. The output arguments that control points will rely on most will be: NrTracks, CurrentURI, CurrentURIMetaData, and MediaDuration.

This also includes the following possibilities and assumptions:

- some output arguments can be progressively updated as the device acquires more information (for example, NrTracks);
- some output arguments can have values to indicate that the information cannot be provided (for example, NOT_IMPLEMENTED value for MediaDuration or CurrentURIMetaData);
- some output arguments can have placeholder values to indicate information is not yet available (for example, DIDL-Lite document with schema-compliant placeholder values).

S	C	DMR	n/a	n/a	[50]

This suggestion permits devices to use placeholder values where appropriate-although the guideline strongly encourages information be as accurate as possible.

7.4.1.6.13 MM GetPositionInfo behaviour

7.4.1.6.13.1 If the AVT.AVTransportURI instance state variable has an empty value, then the AVT:GetPositionInfo must return values in the following manner.

- Track output argument must have a value of "0".
- TrackURI output argument must be "".

M	R	DMR	n/a	n/a	[50]

These guidelines specify the required behaviour for Track and TrackURI output arguments of AVT:GetPositionInfo.

7.4.1.6.13.2 If the AVT.AVTransportURI instance state variable points to content that is neither a media collection file nor a DLNA PlayContainer URI, then the AVT:GetPositionInfo must return values in the following manner.

- The track output argument must have a value of "1".
- The trackURI output argument must be the same as AVT.AVTransportURI.

M	R	DMR	n/a	n/a	[50]

This guideline works in conjunction with 7.4.1.6.11.2.

7.4.1.6.13.3 If the AVT.AVTransportURI instance state variable points to a media collection file or a DLNA PlayContainer URI, then the AVT:GetPositionInfo must return values in the following manner.

- The track output argument must be equal to AVT.CurrentTrack value, as described in 7.4.1.6.11.4.
- The trackURI output argument must be equal to AVT.CurrentTrackURI, as described in 7.4.1.6.11.4.

There may be a time period where the URI and track values are temporarily non-synchronized for transitioning purposes.

M	R	DMR	n/a	n/a	[50]

When the MediaRenderer is in the TRANSITIONING state, the URI values and track index values may be non-synchronized. It is mandatory that when the MediaRenderer enters a non-transitional state (for example, PLAY, STOPPED, etc.), the URI and track values must be accurate as described in the guideline.

7.4.1.6.13.4 The TrackDuration output argument of an AVT:GetPositionInfo response must be one of the following:

- a representation of the total playback duration for the content indicated by TrackURI and AVT.CurrentTrackURI;
- has a value of "NOT_IMPLEMENTED".

M	R	DMR	n/a	n/a	[50]

Provides baseline expectations for reporting the playback duration of the content that is currently rendered. This guideline does not impose any mandatory accuracy requirements because methodologies for determining the total playback duration varies between implementations and often has a dependency on the media formats.

7.4.1.6.13.5 The RelTime and AbsTime output arguments must comply with the following rules:

- the RelTime output argument must report the current playback position for the content indicated by TrackURI. If the value cannot be provided or if a time value is not applicable, then the value must be "NOT_IMPLEMENTED";
- the AbsTime output argument must report the current playback position for content indicated by AVT.AVTransportURI. If the value cannot be provided or if a time value is not applicable, then the value must be "NOT_IMPLEMENTED".

M	R	DMR	n/a	n/a	[50]

This guideline specifies the correct behaviour for reporting the AbsTime and RelTime output arguments. Reporting an empty string for either is not acceptable.

7.4.1.6.13.6 The *TrackMetadata* output argument must be formatted in one of the following ways:

- the value specifies a valid DIDL-Lite document with a single <item> element that describes the content indicated by AVT.CurrentTrackURI and the TrackURI output argument;
- the value specifies "NOT_IMPLEMENTED" to indicate that metadata is not supported or not available.

M	R	DMR	n/a	n/a	[50]

The AVTransport specification requires that the *TrackMetadata* value provide DIDL-Lite metadata or a value of NOT_IMPLEMENTED. Reporting an empty string is not acceptable.

7.4.1.6.13.7 Information returned by AVT:GetPositionInfo should be as accurate as possible. The output arguments that control points will rely on most will be: RelTime, TrackDuration, Track, and TrackURI.

S	C	DMR	n/a	n/a	[50]

This suggestion permits devices to use placeholder values where appropriate -although the guideline strongly encourages information be as accurate as possible.

7.4.1.6.14 MM metadata reporting

7.4.1.6.14.1 If a UPnP MediaRenderer does not support the CurrentURIMetaData input argument, then the MediaRenderer must accept a AVT:SetAVTransportURI request as though an empty value was sent for CurrentURIMetaData.

M	C	DMR	n/a	n/a	[50]

This guideline essentially requires to MediaRenderers to discard the CurrentURIMetaData value if the input argument is not supported. This guideline allows control points to use the CurrentURIMetaData input argument when invoking AVT:SetAVTransportURI, without having to implement logic for retrying the request without metadata.

Note that DLNA interoperability guidelines requirement 7.3.16.1 only requires UPnP devices to accept SOAP requests up to 20 480 bytes (20 KB) in size.

7.4.1.6.14.2 If a UPnP MediaRenderer supports the CurrentURIMetaData input argument (of AVT:SetAVTransportURI), then the MediaRenderer may choose not to validate the metadata specified by the control point.

O	C	DMR	n/a	n/a	[50]

This guideline absolves MediaRenderers from having to parse or validate the DIDL-Lite metadata sent by a control point.

7.4.1.6.14.3 UPnP MediaRenderers that support the CurrentURIMetaData input argument of AVT:SetAVTransportURI must provide the CurrentURIMetaData output argument of the AVT:GetMediaInfo response, using the following rules.

- The value must be a DIDL-Lite document that complies with the restrictions in 7.4.1.6.14.9.

- The value must be an empty string.

M	L	DMR	n/a	n/a	[50]

A MediaRenderer that improperly truncates the DIDL-Lite fragment runs the risk of creating an invalid XML fragment, which can have a seriously negative impact on a control point.

7.4.1.6.14.4 A UPnP AV MediaRenderer must implement the AVT.AVTransportURIMetaData virtual instance state variable, such that the AVT.AVTransportURIMetaData is the same value as the CurrentURIMetaData output argument of AVT:GetMediaInfo.

M	C	DMR	n/a	n/a	[50] [54]

7.4.1.6.14.5 A UPnP AV MediaRenderer must update the AVT.AVTransportURI virtual instance state variable to indicate the current URI prior to changing the AVT.TransportState virtual instance state variable to "PLAYING".

M	L	DMR	n/a	n/a	[50] [54]

This allows control points to know when the MediaRenderer has selected a different <res> element.

Control points can rely on the GENA event to indicate when the DMR has chosen an alternate URI.

This guideline applies generally, including these scenarios:

- the DMR uses the URI specified in the CurrentURI argument of the AVT:SetAVTransportURI request;
- the DMR overrides the specified URI (in the CurrentURI argument) with another one (from the CurrentURIMetaData argument) in the AVT:SetAVTransportURI request (as described in 7.4.1.6.9.1).

7.4.1.6.14.6 UPnP MediaRenderers may specify a different value for

- the AVT.AVTransportURIMetaData instance state variable; and
- the CurrentURIMetaData output argument of AVT:GetMediaInfo

when compared against the CurrentURIMetaData input value of AVT:SetAVTransportURI.

O	C	DMR	n/a	n/a	[50]

MediaRenderers may have different values for a variety of reasons.

- The device may parse different metadata from within the content file.
- The device may remove elements and attributes from the DIDL-Lite document (while remaining schema-compliant) to reduce memory use.
- The device may truncate values of elements and attributes to reduce memory use.

7.4.1.6.14.7 If a UPnP MediaRenderer specifies a different, non-empty value for

- the AVT.AVTransportURIMetaData instance state variable; or

- the CurrentURIMetaData output argument of AVT:GetMediaInfo.

when compared against the CurrentURIMetaData input value of AVT:SetAVTransportURI, then the MediaRenderer must impose the following restrictions on the metadata.

- The provided metadata must represent the metadata of the content indicated by the AVT.AVTransportURI instance state variable (equivalently, the value specified in the CurrentURI argument or a URI obtained from a <res> element in the CurrentURIMetaData argument of the AVT:SetAVTransportURI request).
- If the AVT.AVTransportURI instance state variable points to a media collection, then the provided metadata must be limited to the media collection. The provided metadata must not include metadata for items within the media collection.

M	L	DMR	n/a	n/a	[50]

Many control points have problems accepting large metadata values. This guideline limits the metadata for AVT.AVTransportURIMetaData and CurrentURIMetaData to refer specifically to the content indicated by the AVT.AVTransportURI instance state variable. This guideline prohibits the general practice of collectively using the metadata of each item in a media collection to represent the metadata of a media collection.

7.4.1.6.14.8 If a UPnP MediaRenderer is capable of playing a media collection and if the MediaRenderer has access to metadata of individual items in the collection, then the MediaRenderer should report that metadata through the AVT.CurrentTrackMetadata instance state variable and the TrackMetadata output argument of AVT:GetPositionInfo.

S	C	DMR	n/a	n/a	[50]

Control points often use metadata provided by the MediaRenderer in user interfaces. MediaRenderers are strongly encouraged to provide such metadata whenever possible.

7.4.1.6.14.9 If a UPnP MediaRenderer control point specifies a value for the CurrentURIMetaData argument of an AVT:SetAVTransportURI request, then the control point must follow these restrictions for the value of the CurrentURIMetaData argument:

- compliant with the DIDL-Lite schema;
- exactly one <DIDL-Lite> element;
- exactly one <item> or <container> element;
- exactly one <dc:title> element and value;
- a minimum of zero and a maximum of one <dc:creator> element and value;
- exactly one <upnp:class> element and value;
- a minimum of one <res> element.

All other XML elements are permitted as long as they are properly declared with their namespaces.

The provided metadata must represent the metadata of the content indicated by the CurrentURI input argument.

One of the <res> elements must be the <res> element that contains the URI specified in the CurrentURI input argument.

If the CurrentURI input argument points to a media collection, then the provided metadata must be limited to the media collection. The provided metadata must not include metadata for items within the media collection.

M	L	DMC +PU+	M-DMC	MIU	[50]

This guideline limits the scope of the CurrentURIMetaData to the metadata directly associated with the CurrentURI input argument. Metadata for a media collection is permitted, so long as value does not provide metadata for each of the individual items in the collection.

Many MediaRenderers have problems storing a large amount of metadata provided in the CurrentURIMetaData. Equally problematic is the fact that many control points cannot support a scenario where a UPnP device transmits a UPnP event or an AVT:GetMediaInfo response with a large amount of metadata.

The expected metadata to be sent in the CurrentURIMetaData argument is best described by the CDS:Browse response for the following request:

- ObjectID: The CDS object ID of that provided the URI specified in the CurrentURI input argument of AVT:SetAVTransportURI;
- BrowseFlag: BrowseMetadata;
- Filter: One or more of the following: ALLIP, res (and or any res attribute), dc:creator, and any other metadata that the control point wants to provide.

Whenever possible, control points should provide all of the available <res> attributes that are normative for DIDL-Lite. Likewise, MediaRenderers should accept and preserve these attributes.

Lastly the guideline permits control points to specify a single <res> element in the metadata, on behalf of a user request. In such cases, the CurrentURIMetaData argument only includes the <res> element that corresponds to the URI specified in the CurrentURI argument.

However, control points are encouraged to provide all available <res> elements. This allows MediaRenderers the opportunity to choose a <res> element that might provide a better rendering experience. See 7.4.1.6.9 for more information about MediaRenderers that select alternate URIs from the CurrentURIMetaData argument.

7.4.1.6.14.10 Control points that receive metadata from a UPnP MediaRenderer must be tolerant of DIDL-Lite metadata that is valid and conformant to DLNA restrictions.

Tolerant behaviour is defined as being able to parse-and-accept or parse-and-ignore the metadata. Failing to parse a DLNA-compliant UPnP action response or event because of metadata is unacceptable behaviour.

M	C	DMC +PU+	M-DMC	MIU	[50]

Control points that invoke UPnP actions or subscribe to UPnP events that involve metadata must be prepared for the presence of metadata, even if metadata is not always provided.

7.4.1.6.14.11 Control points that receive metadata from a UPnP MediaRenderer should be tolerant of DIDL-Lite metadata that is invalid or not conformant to DLNA restrictions.

Tolerant behaviour is defined as being able to parse-and-ignore the metadata.

S	C	DMC +PU+	M-DMC	MIU	[50]

Control points that can handle DIDL-Lite metadata that is not schema compliant exhibit a good level of robustness in an environment where non-DLNA control points may be interacting with the MediaRenderer device.

7.4.1.6.15 MM reporting transport information

7.4.1.6.15.1 UPnP MediaRenderers that respond to an AVT:GetTransportInfo request must reflect the play/transport state in the following manner.

- The CurrentTransportState output argument must match the AVT.TransportState instance state variable.
- The CurrentTransportStatus output argument must match the AVT.TransportStatus instance state variable.
- The CurrentSpeed output argument must match the AVT.TransportPlaySpeed instance state variable.

M	R	DMR	n/a	n/a	[50]

This guideline makes it so that control points can safely assume that UPnP actions and instance state variables report the same transport state information.

7.4.1.6.15.2 UPnP MediaRenderers that respond to an AVT:GetTransportSettings request must accurately reflect the PlayMode output argument to match the AVT.CurrentPlayMode instance state variable.

M	R	DMR	n/a	n/a	[50]

7.4.1.6.16 MM normative MediaRenderer state transitions

7.4.1.6.16.1 If a UPnP MediaRenderer enters the TRANSITIONING state, it must change to the state desired by the control point within 30 s.

The longest period of time that a MediaRenderer device is permitted to remain in the TRANSITIONING state is 30 s.

M	L	DMR	n/a	n/a	[50]

The TRANSITIONING state is a way for a device to indicate that it is attempting to change into a different state, such as PLAYING or STOPPED.

7.4.1.6.16.2 UPnP MediaRenderers may enter the TRANSITIONING state at any time.

O	C	DMR	n/a	n/a	[50]

The AVTransport specification has an informative diagram that describes TRANSITIONING to be used only between STOPPED and PLAYING states. This diagram is not restrictive, and it allows new transitions.

The TRANSITIONING provides a useful cue to the user that the device is trying to do something. For example, entering the TRANSITIONING state after a call to AVT:SetAVTransportURI acknowledges the user's request to change content even if playback has not yet begun. Likewise, if a device is in the PLAYING state and network problems interrupt playback, the device can go into the TRANSITIONING state during the interruption.

7.4.1.6.16.3 UPnP MediaRenderers must not define a new intermediate state.

An intermediate state is a state that the device enters temporarily before entering the state requested by the control point.

M	L	DMR	n/a	n/a	[50]

Defining a new intermediate state can confuse control points into believing other control points are trying to use the device. A control point that encounters a vendor-defined state of BUFFERING has no idea that this state is an intermediate state. Likewise, a control point that encounters a vendor-defined state of LOCKED has no idea that the device may remain in this state indefinitely. This guideline allows control points to always assume that vendor-defined states can last indefinitely and that TRANSITIONING is the only intermediate state.

7.4.1.6.16.4 UPnP MediaRenderers may define new non-intermediate states.

O	L	DMR	n/a	n/a	[50]

For example, defining a LOCKED state to indicate that the DMR cannot accept AVTransport requests is acceptable due to its current state (for example, a local user is using the device directly).

However, creating a new state called BUFFERING to represent that the device is busy buffering data in preparation for rendering is not permitted.

Note that the LOCKED and BUFFERING states are only examples. The key distinction between the examples is the former which involves an out-of-scope scenario and the latter involves an in-scope scenario. In the case of LOCKED, an external stimulus (i.e. out-of-scope scenario) caused the device to enter the LOCKED state. In the case of BUFFERING, the DMR entered the BUFFERING state after a control point requested a normative state change request (for example, play state change, track change, URI change, etc.).

7.4.1.6.16.5 UPnP MediaRenderers may define new allowed values for the AVT.TransportStatus instance state variable.

O	C	DMR	n/a	n/a	[50]

In lieu of defining new intermediate states, vendors are permitted to use the TransportStatus instance state variable to convey additional information about the transport layer.

7.4.1.6.16.6 UPnP MediaRenderers that define new allowed values for the AVT.TransportStatus instance state variable should begin the allowed value with "ERROR_" to indicate the value represents an error condition.

S	L	DMR	n/a	n/a	[50]

This guideline allows control points to mark status values as being informative or error-related. The normative error value for AVT.TransportStatus is ERROR_OCCURRED.

7.4.1.6.17 MM transport actions

7.4.1.6.17.1 The comma-separated list of action names returned in AVT:GetCurrentTransportActions may change depending on what the device is doing and what content the device is accessing.

O	R	DMR	n/a	n/a	[50]

These guidelines reference the permitted and expected behaviours of a device that dynamically enables/disables its AVTransport actions.

7.4.1.6.17.2 The value returned in the Actions output argument of an AVT:GetCurrentTransportActions request must match the value of the AVT.CurrentTransportActions instance state variable.

M	R	DMR	n/a	n/a	[50]

7.4.1.6.17.3 A MediaRenderer control point should provide a UI indicator to inform a user of disabled transport actions.

S	C	DMC +PU+	M-DMC	n/a	n/a

This guideline requires control points to provide UI indications to inform users that a playback feature is disabled. Without this information, a user can be misled into believing that the device is not working properly. Examples of user indicators include the following:

- gray-out the button for the disabled operation;
- an icon that flashes on the screen to indicated the disabled state when the user pushes the button.

Depending on UI form factor this guideline may be difficult to implement, such as a handheld remote with physical buttons.

7.4.1.6.17.4 A UPnP MediaRenderer must report the disabling of the Pause operation by excluding the AVT:Pause from the Actions output argument of an AVT:GetCurrentTransportActions and the AVT.CurrentTransportActions instance state variable.

M	C	DMR	n/a	n/a	n/a

7.4.1.6.17.5 If a UPnP MediaRenderer's AVT.NumberOfTracks value is greater than 1, then the AVT:Next/AVT:Previous actions must have the behaviour of incrementing/decrementing the AVT.CurrentTrack instance state variable (and likewise properly reflecting other state change information required by the AVTransport specification).

M	R	DMR	n/a	n/a	[50]

This guideline defines the behaviour of AVT:Next and AVT:Previous in the context of playing a media collection. Behaviour for AVT:Next and AVT:Previous is not defined for other cases.

7.4.1.6.18 MM play mode behaviour

7.4.1.6.18.1 UPnP MediaRenderers that implement AVT:SetPlayMode must implement the method such that changes to the current play mode are applied immediately.

M	C	DMR	n/a	n/a	[50]

An example of bad behaviour is a MediaRenderer that requires a control point to invoke AVT:Play after a call to AVT:SetPlayMode in order for the requested play mode to be applied.

7.4.1.6.18.2 UPnP MediaRenders that change the play mode may change the transport state so long as the new state is not STOPPED.

O	C	DMR	n/a	n/a	[50]

7.4.1.6.18.3 UPnP MediaRenderers should keep the same play mode after a call to AVT:SetAVTransportURI.

S	C	DMR	n/a	n/a	[50]

Automatically changing unrelated portion of the device's state is not a good practice.

7.4.1.6.19 MM play modes

7.4.1.6.19.1 UPnP MediaRenderers that implement the "NORMAL" play mode must implement it in the following manner:

- an AVT:Next request results in AVT.CurrentTrack being incremented by one;
- AVT:Next requests that attempt to change the track number beyond the last track must result with no state change (i.e. request accepted and ignored) or a response with a UPnP AV error code 711 (illegal seek target);
- an AVT:Previous request results in AVT.CurrentTrack being decremented by one;
- AVT:Previous requests that attempt to change the track number before the first track must result with no state change (i.e. request accepted and ignored) or a UPnP AV error code 711 (illegal seek target);
- if a new value is applied to AVT.CurrentTrack, then AVT.CurrentTrackURI is updated appropriately;
- if the play state before an AVT:Next or AVT:Previous request is PLAYING and a new track is applied, then the device continues playback with the new track;
- if the device is in the PLAYING state, with AVT.CurrentTrack = AVT.NumberOfTracks, and playback finishes for the content indicated by AVT.CurrentTrackURI, then the MediaRenderer enters the STOPPED state and AVT.CurrentTrack is reset to 1. AVT.CurrentTrackURI is appropriately updated, but AVT.AVTransportURI remains the same. This bulleted item does not apply when AVT.NextAVTransportURI is set as a result of AVT:SetNextAVTransportURI.

M	C	DMR	n/a	n/a	[50]

The AVTransport specification does not specify the behaviour of different play modes. These guidelines specify the basic expectations, inspired largely from traditional consumer electronic devices that play optical media content. The AVTransport specification may require additional behaviours of the MediaRenderer.

7.4.1.6.19.2 UPnP MediaRenderers that implement the "REPEAT_ONE" play mode must implement it in the same manner as 7.4.1.6.19.1, except in the following.

- If the device is in the PLAYING state and playback reaches the end for the content indicated by AVT.CurrentTrackURI, then the MediaRenderer changes the playback position to "00:00:00" and continues playing the same content from the beginning.

M	C	DMR	n/a	n/a	[50]

7.4.1.6.19.3 UPnP MediaRenderers that implement the "REPEAT_ALL" play mode must implement it in the same manner as 7.4.1.6.19.1, with the following exceptions:

- AVT:Next requests that attempt to change the track number beyond the last track must result with AVT.CurrentTrack set to "1".
- AVT:Previous requests that attempt to change the track number before the first track must result with AVT.CurrentTrack set to AVT.NumberOfTracks.
- If the device is in the PLAYING state, with AVT.CurrentTrack = AVT.NumberOfTracks, and playback finishes for the content indicated by AVT.CurrentTrackURI, then the MediaRenderer resets AVT.CurrentTrack to 1 and AVT.CurrentTrackURI is appropriately updated. AVT.AVTransportURI remains unchanged and content playback continues with the new track.

M	C	DMR	n/a	n/a	[50]

7.4.1.6.19.4 UPnP MediaRenderers that implement the "RANDOM" play mode must implement it in the following manner:

- AVT:Next and AVT:Previous requests must result with AVT.CurrentTrack set to a random value from "1" to AVT.NumberOfTracks, with the AVT.CurrentTrackURI getting updated appropriately. If the play state is PLAYING, then content playback continues with the new track;
- if a new value is applied to AVT.CurrentTrack, then AVT.CurrentTrackURI getting updated appropriately;
- if the play state before an AVT:Next or AVT:Previous request is PLAYING and a new track is applied, then the device continues playback with the new track;
- if the device is in the PLAYING state and playback finishes for the content indicated by AVT.CurrentTrackURI, then the MediaRenderer sets a new random value for AVT.CurrentTrack and AVT.CurrentTrackURI is appropriately updated. AVT.AVTransportURI remains unchanged and content playback continues with the new track.

M	C	DMR	n/a	n/a	[50]

7.4.1.6.19.5 UPnP MediaRenderers that implement the "SHUFFLE" play mode must implement it in the same manner as "RANDOM" (see 7.4.1.6.19.4) except for the following manners:

- the device must track the value history of AVT.CurrentTrack so that the new track value is not a repeat of a previously played track;
- when the MediaRenderer has played all of the items (i.e. all tracks have been played), then the device enters the STOPPED state and the AVT.CurrentTrack changes to "1".

M	C	DMR	n/a	n/a	[50]

7.4.1.6.19.6 UPnP MediaRenderers should support the "REPEAT_ONE", "REPEAT_ALL" and either "SHUFFLE" or "RANDOM" play modes.

S	L	DMR	n/a	n/a	[50]

The "NORMAL" play mode is required, but these additional play modes are recommended.

7.4.1.6.20 MM play speed

UPnP MediaRenderers must specify their allowed values for AVT.TransportPlaySpeed in the service description as a rational fraction, as required by the AVTransport specification.

M	R	DMR	n/a	n/a	[50]

Specifying allowed values like "NORMAL", "2x" or "Backwards Slow 0,25" are not allowed. Examples of correct representations are "1", "2", and "-1/4".

This guideline imposes no requirement on a control point to represent play speeds on a user interface as a rational fraction.

7.4.1.6.21 MM renderer volume control

7.4.1.6.21.1 UPnP MediaRenderers that support volume control must implement the RCS.Volume instance state variable with a range of 0 to 100, where 0 is audibly equivalent to mute and 100 is the maximum loudness. The stepping for the variable must be 1.

A MediaRenderer implements volume control when it implements the RCS.Volume instance state variable.

M	L	DMR	n/a	n/a	[60]

This guideline provides baseline expectations on how a control point can change the volume on a MediaRenderer.

The stepping of the RCS.Volume value does not have to correspond to that of actual sound volume. For example the actual heard volume stepping may change per five steppings of the RCS.Volume instance state variable.

7.4.1.6.21.2 UPnP MediaRenderers that support volume control must implement RCS:SetVolume, RCS:GetVolume, RCS:SetMute, and RCS:GetMute.

M	L	DMR	n/a	n/a	[60]

7.4.1.6.22 MM DLNAQOS support

If DLNAQOS as defined in 7.2 is implemented, AVTransport SOAP actions must be tagged with DLNAQOS_2, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), for both requests and responses in accordance with Table 8.

M	A	DMR DMC +PU+	M-DMC	MIU	n/a

All other forms of UPnP AV traffic described in this subclause (for example, CDS, rendering service) are tagged according to the default DLNAQOS_UP value in Table 8.

7.4.1.7 Upload and optional content management requirements

7.4.1.7.1 MM/CM – DMS with upload device option support definition

7.4.1.7.1.1 A UPnP AV MediaServer may support the upload device option by implementing the baseline upload AnyContainer (defined in 7.4.1.7.11) and optionally the optional content management operations (OCM operations, as defined in 7.4.1.7.2).

O	A	DMS	M-DMS	n/a	[52] [55]

This guideline means that a DMS or M-DMS may implement the Upload Device Option (i.e. can receive uploaded content from an +UP+ or M-DMU).

If the DMS or M-DMS implements the Upload Device Option, then it must implement upload AnyContainer. It may additionally support various OCM operations.

7.4.1.7.1.2 If a UPnP AV MediaServer supports the upload device option, it must support the upload AnyContainer operation.

M	A	DMS	M-DMS	n/a	[52] [55]

The baseline requirement for a DMS or M-DMS that supports the upload device option is to be able to receive a CDS:CreateObject request that specifies the "DLNA.ORG_AnyContainer" as the parent container for a new CDS item. The DMS or M-DMS must be able to receive the content through an HTTP POST request.

The following is an example sequence of events for an upload scenario.

- The upload controller invokes CDS:CreateObject on the DMS. The metadata describes an image to be created in "DLNA.ORG_AnyContainer".
- The DMS approves the metadata from the CDS:CreateObject request to determine if it is valid. Since the parent container specifies "DLNA.ORG_AnyContainer", then the DMS decides that the new image object belongs in an existing CDS container (title of "New Photos") for all image uploads.
- The DMS sends the CDS:CreateObject response to the Upload Controller. The response indicates the object will be in the "New Photos" container. The response also includes a <res> element that omits a URI value but has a URI value for res@importUri.
- The Upload Controller uses an HTTP POST request to transfer the image file to the DMS.

When the DMS is able to serve the new image, it provides a URI value for the <res> element.

7.4.1.7.2 MM/CM – Optional content management operation definitions

If a UPnP AV MediaServer supports optional content management (OCM) operations, it may support one or more of the following OCM operations.

- OCM: upload content Use this to upload content to a specific CDS container. This operation has 2 steps: use CDS:CreateObject to create a CDS item and use HTTP POST to transfer the content.

- **OCM: create child container** Use this to create a new CDS container in a specified CDS container. This operation has one step: use CDS:CreateObject to create a CDS container. This operation used generally as a preceding step to an OCM: upload content operation.
- **OCM: destroy item** Use this to destroy a CDS item. This operation has one step: use CDS:DestroyObject to destroy a CDS item.

O	A	DMS	M-DMS	n/a	[52] [55]

The guidelines define interoperability specifically for the described usages. Vendors must always implement behaviour that is consistent with the DLNA guidelines, even if the implementation will be used for an operation that has different preconditions. For example a DMS that supports the OCM: create child container may allow control points to create CDS containers. In such a scenario, the DMS and control point need to abide by appropriate syntax rules for CDS:CreateObject.

The guideline also permits other forms of management operations, although the guidelines do not define interoperability rules for them.

7.4.1.7.3 MM/CM – Upload controller and mobile digital media uploader

7.4.1.7.3.1 A DLNA device class may implement the upload controller device capability.

O	A	DMS DMP DMR DMC DMPr	M-DMS M-DMP M- DMC M-DMU	n/a	[52] [55]

7.4.1.7.3.2 An upload controller or a mobile digital media uploader must implement a UPnP AV MediaServer control point capable of invoking the following actions.

- CDS:CreateObject.

By supporting these actions, an upload controller must be capable of supporting either the upload AnyContainer operation or the OCM: upload content operation.

M	C	+UP+	M-DMU	n/a	[52] [55]

CDS:CreateObject is the action that allows an Upload Controller or M-DMU to upload content. The guidelines specify that the normative content transfer methodology involves HTTP POST, as described in guideline 7.5.5.6.1.1

7.4.1.7.3.3 An upload controller or M-DMU must support the upload AnyContainer operation.

M	C	+UP+	M-DMU	n/a	[52] [55]

7.4.1.7.3.4 An upload controller or M-DMU may implement a UPnP AV MediaServer control point capable of invoking the following actions.

- CDS:DestroyObject

By supporting these actions, an upload controller may be capable of supporting additional optional content management operations.

See 7.4.1.7.2 for more information about optional content management operations.

O	C	+UP+	M-DMU	n/a	[52] [55]

CDS:DestroyObject is the action that allows an upload controller to destroy a CDS object when a content transfer failed or when the user wants to remove uploaded content from the DMS.

Upload Controllers and M-DMUs are allowed to implement support for other CDS actions, but the DLNA guidelines do not specify the interoperability behaviour for other actions.

7.4.1.7.4 MM/CM – Determining upload AnyContainer support

7.4.1.7.4.1 If a UPnP MediaServer supports the upload AnyContainer operation or OCM: create child container, then it must use the <dlina:X_DLNA CAP> element (as a child of the <device> element that represents the MediaServer) in the device description document and use the following Capability IDs in the element's comma-separated value list to indicate support for uploading a media class.

Table 18 – Capability IDs for AnyContainer support

Capability ID	Description
audio-upload	The UPnP AV MediaServer supports the upload AnyContainer operation for the Audio media class
image-upload	The UPnP AV MediaServer supports the upload AnyContainer operation for the image media class
av-upload	The UPnP AV MediaServer supports the upload AnyContainer operation for the AV media class
create-child-container	The UPnP AV MediaServer supports the OCM: create child container operation

M	A	DMS	M-DMS	n/a	[52] [55]

DMS devices use the <dlina:X_DLNA CAP> element to indicate support for the upload AnyContainer operation. The element is a comma separated value list that indicates whether the DMS can receive uploads of images, audio-only, or audio/video content.

It should be noted that a DMS that supports the upload AnyContainer operation is different from a DMS with an Upload Controller. The former is a DMS that can receive uploaded content. The latter is a DMS that can upload to a different DMS. It is possible to implement a DMS that supports the upload AnyContainer operation and the Upload Controller capability.

A DMS that supports OCM: create child container must also support the creation of child containers where the DMS chooses the parent container (because the upload controller specified DLNA.ORG_AnyContainer as the parent). This guideline explains how upload controllers determine if OCM: create child container is supported for the DMS.

See guideline 7.3.36 for the formal syntax of the <dlina:X_DLNA CAP> element.

7.4.1.7.4.2 A UPnP AV MediaServer may define the media format profiles that it will accept in the CDS:CreateObject action through the use of the CDS:X_GetDLNAUploadProfiles action.

O	A	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.4.3 The CDS:X_GetDLNAUploadProfiles action's definition in the service description must be defined as indicated below.

```
<action>
  <name>X_GetDLNAUploadProfiles</name>
  <argumentList>
    <argument>
      <name>UploadProfiles</name>
      <direction>in</direction>
      <relatedStateVariable>X_A_ARG_Type_UploadProfiles</relatedStateVariable>
    </argument>
    <argument>
      <name>SupportedUploadProfiles</name>
      <direction>out</direction>
      <relatedStateVariable>X_A_ARG_Type_SupportedUploadProfiles</relatedStateVariable>
    </argument>
  </argumentList>
</action>
```

The X_A_ARG_TYPE_UploadProfiles and X_A_ARG_Type_SupportedUploadProfiles state variables are defined below.

```
<stateVariable sendEvents="no">
  <name>X_A_ARG_Type_UploadProfiles</name>
  <dataType>string</dataType>
</stateVariable>
<stateVariable sendEvents="no">
  <name>X_A_ARG_Type_SupportedUploadProfiles</name>
  <dataType>string</dataType>
</stateVariable>
```

M	A	DMS	M-DMS	n/a	[52] [55]

The UploadProfiles input argument is an unordered, comma separated list of DLNA media format profile names.

The SupportedUploadProfiles output argument is an unordered, comma-separated list of DLNA media format profile names, as described below:

- are listed in the UploadProfiles input argument and this MediaServer is willing to accept at the time the action is invoked;
- or, in case of empty UploadProfiles input argument, the SupportedUploadProfiles list must contain the complete list of DLNA media format profiles that this MediaServer is willing to accept at the time the action is invoked.

The DLNA media profile IDs that appear in SupportedUploadProfiles must comply with these restrictions:

- must be AV, audio, or image media classes;
- media format profile IDs for icons, thumbnails, media collection files, and XHTML print documents are expressly prohibited.

The response behaviour is summarized in the following way:

- if UploadProfiles is empty, then SupportedUploadProfiles contains a complete list of profiles that the MediaServer is willing to accept at the current time. Control points specify an empty value for UploadProfiles when they want to get a full list of profiles that the MediaServer will accept for uploads;
- if UploadProfiles contains one or more profiles, then SupportedUploadProfiles contains the subset of UploadProfiles that the MediaServer is willing to accept at the current time. Control points specify one or more profiles for UploadProfiles when they are interested in uploading specific formats to a MediaServer.

7.4.1.7.5 MM/CM: Operations that need CDS:CreateObject

If a UPnP AV MediaServer supports one or more of these operations, then it must implement CDS:CreateObject.

- upload AnyContainer operation;
- OCM: upload content;
- OCM: create child container.

M	A	DMS	M-DMS	n/a	[52] [55]

The CDS:CreateObject action is used to create a CDS object that will represent the uploaded content.

In addition to these guidelines, a DMS or M-DMS with the ability to receive uploaded content must implement an HTTP server capable of processing HTTP POST requests, as described in 7.5.5.6.1.

7.4.1.7.6 MM/CM: Operations that need CDS:DestroyObject

If a UPnP AV MediaServer supports this operation, then it must implement CDS:DestroyObject.

- OCM: destroy item

M	A	DMS	M-DMS	n/a	[52] [55]

The CDS:DestroyObject action is used for a variety of OCM operations related to removing CDS objects from a DMS.

7.4.1.7.7 MM/CM – Other CDS actions

A UPnP AV MediaServer may implement CDS:UpdateObject, CDS:DeleteResource, CDS:CreateReference, CDS:ImportResource, CDS:ExportResource, or CDS:StopTransferResource.

O	A	DMS	M-DMS	n/a	[52] [55]

These are normative UPnP AV CDS actions, but the DLNA guidelines do not define interoperability rules for them.

7.4.1.7.8 MM/CM: Baseline media formats

A UPnP AV MediaServer that belongs to one of the DLNA-defined Device Categories and implements the upload AnyContainer operation for a DLNA Media Class (as indicated by guideline 7.4.1.7.4.1) must support the uploading of the mandatory Media Format Profiles for that DLNA Media Class in that Device Category.

M	A	DMS	M-DMS	n/a	[52] [55]

DLNA guidelines have an expressed goal to facilitate uploading of baseline media formats. A DMS or M-DMS that only supports uploads of optional media formats detracts from the guidelines' interoperability message.

7.4.1.7.9 MM/CM – Indicating support for OCM operations

7.4.1.7.9.1 If a UPnP AV MediaServer supports one or more OCM operations, then the UPnP AV MediaServer may have one or more CDS objects with the @dlna:dlnaManaged attribute.

O	A	DMS	M-DMS	n/a	[52] [55]

The @dlna:dlnaManaged attribute indicates the OCM operations that the DMS or M-DMS is able to support for a given CDS object.

7.4.1.7.9.2 If a UPnP AV MediaServer supports one or more OCM operations on a CDS object, then the UPnP AV MediaServer must use the @dlna:dlnaManaged attribute to indicate support for those OCM operations on a CDS object.

M	A	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.9.3 If a CDS object allows one or more OCM operations, then the CDS object must have a @restricted value of "0". Likewise, a @restricted value of "1" must be used when the CDS object does not support any OCM operations.

M	A	DMS	M-DMS	n/a	[52] [55]

DLNA defines a restriction that a @restricted value of "1" is used when the CDS object supports no OCM operations.

7.4.1.7.9.4 The @dlna:dlnaManaged attribute is a 32-bit unsigned integer encoded into exactly 8 hexadecimal digits, with the following bit definitions. Bit-0 is the least significant bit. If a bit supports a particular operation, then the bit value is true. Otherwise, the bit value is false to indicate the operation is not supported. (for example, 00000000000000000000000000000001b = 0x00000001 where bit-0 is set to true)

The hexadecimal encoded form must consist only of hexadecimal digits. The value must omit the "0x" string that often precedes hexadecimal notation:

- Bit-0: indicates support for OCM: upload content;
- if true then the MediaServer allows a control point to create child CDS items in the container for the OCM: upload content operation;
- must be false when used with a CDS item;
- Bit-1: indicates support for OCM: create child container;
- if true on a CDS container, then the MediaServer allows a control point to create child CDS containers that can support the OCM: upload content;
- must be false when used with a CDS item;
- Bit-2: indicates support for OCM: destroy item;

- if true on a CDS item, then it means the CDS item must be removed from the CDS hierarchy when the CDS object is successfully destroyed through an OCM: destroy item request;
- must be false when used with a CDS container;
- all other bits must be false. All other bits are reserved for future use.

M	A	DMS	M-DMS	n/a	[52] [55]

This guideline defines the syntax for the @dlna:dlnaManaged value. This attribute describes the DMS implementation's ability to support operations that affect the CDS object as a whole.

It should be noted that a false value for a bit means that the DMS or M-DMS does not claim support for the described operation. Control points are expected to honor the interpretation of these bits to maximize interoperability because invoking a CDS action that is related to an unsupported OCM operation are likely to receive an error. CDS failure responses are not mandatory when a DMS or M-DMS detects a deviation because vendors are permitted to use normative CDS actions for vendor-defined operation.

There may be cases where a DMS (that is normally able to destroy the CDS item) is not able to destroy a CDS item at the time of the request. For example, the actual content binary file might be locked or a local management policy prevents the CDS item from being destroyed at the current moment.

7.4.1.7.10 MM/CM – Parallel upload AnyContainer and OCM operations

7.4.1.7.10.1 If a MediaServer control point attempts to do multiple upload AnyContainer or multiple OCM operations in parallel, and if the MediaServer fails one or more of the parallel attempts, then the MediaServer control point must retry the failed attempts in a serialized manner.

M	A	UP+	M-DMU	MIU	[52] [55]

DMS devices are not required to support parallel attempts to upload or destroy content. Control points that attempt to do so are responsible for retrying in a serialized manner in the event of a failure.

7.4.1.7.10.2 If a UPnP AV MediaServer supports the upload AnyContainer or other OCM operations, then it must be capable of performing at least one upload AnyContainer or OCM operation at a time.

M	A	DMS	M-DMS	MIU	[52] [55]

7.4.1.7.11 MM/CM – Upload AnyContainer operation

7.4.1.7.11.1 If a UPnP AV MediaServer supports the upload AnyContainer operation, then it must allow control points to specify a "DLNA.ORG_AnyContainer" value for the ContainerID input argument in a CDS:CreateObject request.

M	A	DMS	M-DMS	n/a	[52] [55]

The "DLNA.ORG_AnyContainer" allows an upload controller to upload content to a UPnP AV MediaServer without having specific knowledge of where the content should be listed in the CDS hierarchy.

Where the MediaServer creates the new object is dependent on the MediaServer implementation.

The "DLNA.ORG_AnyContainer" is a magic container ID which is used only in the request and a container of this container ID does actually not exist.

Note that a control point can attempt to deviate slightly from the restrictions listed in 7.4.1.7.11, but the result may be an error. For example, if the upload controller tries to use object.item.audioItem.audioBroadcast instead of object.item.audioItem, then the DMS can choose to fail the request.

7.4.1.7.11.2 If a UPnP AV MediaServer supports the upload AnyContainer operation, then the @id value for each container must be a value that is not equal to "DLNA.ORG_AnyContainer".

M	A	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.11.3 If a UPnP AV MediaServer control point is going to start an upload AnyContainer operation, then it must invoke CDS:CreateObject with the following rules:

- the ContainerID input argument must be "DLNA.ORG_AnyContainer";
- the elements input argument must specify a CDS item with a single <res> element that does not have a URI value. The <res> element must also conform to guideline 7.4.1.7.19. The <item> element must also have a <upnp:class> value (or similarly derived value) that corresponds to the media class of the content that is going to be uploaded.

Table 19 – Required media class UPnP values

Media class	Required UPnP:class value
Audio	object.item.audioItem
Image	object.item.imageItem
AV	object.item.videoItem

M	A	+UP+	M-DMU	MIU	[52] [55]

7.4.1.7.11.4 If a UPnP AV MediaServer receives an upload AnyContainer request with a <upnp:class> value that is derived from a supported base class value, then the DMS may change the <upnp:class> value to the supported base class or to a similarly derived class, as indicated in the Result output argument of the CDS:CreateObject.

For example, if the request specifies object.item.audioItem.audioBroadcast, then the MediaServer can change the value to object.item.audioItem. Likewise, if the request specified object.item.imageItem, the MediaServer can change it to object.item.imageItem.photo.

O	C	DMS	M-DMS	n/a	[52] [55]

DMS devices are not required to support the full range of derived values for the <upnp:class> element.

7.4.1.7.11.5 If a UPnP AV MediaServer returns a success response to an Upload Controller's request to start an upload AnyContainer operation, then the MediaServer must do the following:

- the DMS or M-DMS device must determine an appropriate CDS container where the new CDS object will be created;
- the MediaServer must specify the object ID of the parent container as the <item> element's @parentID attribute value, which is returned in the Result output argument;
- the <res> element (found in the Result output argument) that provides the res@importUri value intended for the content transfer process must comply with the guidelines in 7.4.1.7.19.2.

M	A	DMS	M-DMS	n/a	[52] [55]

This guideline describes the proper DMS or M-DMS behaviour in a success scenario. The DMS needs to use an appropriate DMS error in the case of a failure.

7.4.1.7.11.6 A UPnP AV MediaServer that supports the upload AnyContainer operation must be capable of participating in a content transfer process, as described in 7.4.1.7.26.

M	A	DMS	M-DMS	n/a	[52] [55]

These guidelines introduce the second step to the upload AnyContainer operation.

7.4.1.7.11.7 A UPnP AV MediaServer control point that supports the upload AnyContainer operation must be capable of participating in a content transfer process, as described in guidelines 7.4.1.7.26.

M	A	+UP+	M-DMU	MIU	[52] [55]

7.4.1.7.11.8 A UPnP AV MediaServer that creates a new CDS item in an upload AnyContainer operation may create the CDS item in a CDS container with the @dlna:dlnaManaged attribute.

O	C	DMS	M-DMS	n/a	[52] [55]

DMS or M-DMS devices can create the new CDS item in a CDS container that has the @dlna:dlnaManaged attribute.

7.4.1.7.11.9 A UPnP AV MediaServer control point must tolerate scenarios where the MediaServer changes values to metadata properties specified in a CDS:CreateObject request.

Tolerate means that the UPnP AV MediaServer control point is able to complete the necessary content transfer process (including the transfer of IFO files, if necessary) after creating the CDS object.

M	C	+UP+	M-DMU	MIU	[52] [55]

The following are some examples of what a MediaServer is permitted to do:

- the @parentID changes from "DLNA.ORG_AnyContainer" to a different @parentID value;
- the <upnp:class> value changes to a derived class or to a base class of the current value;
- <dc:title>, <dc:creator>, and other string-based user-informational metadata properties are truncated to fit the maximum length supported by the MediaServer.

7.4.1.7.11.10 A UPnP AV MediaServer that creates a new CDS item in an upload AnyContainer operation may change metadata values to indicate the support or unsupported nature of OCM operations.

O	C	DMS	M-DMS	n/a	[52] [55]

MediaServers are permitted to change metadata values at any time. This behaviour is also permissible when the MediaServer actually creates the new CDS object.

7.4.1.7.12 MM/CM: OCM:Upload content operation

7.4.1.7.12.1 If a UPnP AV MediaServer supports the OCM: upload content operation on a CDS container, then it must specify one or more <upnp:createClass> elements to indicate the types of CDS objects that can be created in the container. The values of these upnp:createClass elements must be equal to or derived from object.item.

M	R	DMS	M-DMS	n/a	[52] [55]

This guideline ensures that control points will know what type of CDS objects can be created in the container.

For example, if a container supports the creation of image and audio-only objects, then it would use a <upnp:createClass> element with object.item.imageItem and a <upnp:createClass> element with object.item.audioItem.

Please note that the CDS container may have <upnp:createClass> that are derived from object.container, as described in 7.4.1.7.13.

7.4.1.7.12.2 If a UPnP AV MediaServer control point is going to start an OCM: upload content operation, then it must invoke CDS:CreateObject with the following rules:

- the ContainerID input argument must indicate a CDS container that supports the OCM: upload content operation;
- the elements input argument must be a DIDL-Lite XML fragment that has a CDS item whose <upnp:class> value matches one of the values in the set of <upnp:createClass> elements associated with the CDS container identified by the ContainerID input argument. If the upnp:createClass@includeDerived attribute has a value of "1", then the <upnp:class> value also matches against the classes derived from the <upnp:createClass> value. The <item> element must also have a single <res> element that does not specify a URI value. The <res> element must also conform with guidelines in 7.4.1.7.19.

M	L	+UP+	M-DMU	MIU	[52] [55]

If a MediaServer supports the operation, Upload Controllers and M-DMUs can choose to upload content to specific CDS containers using the OCM: upload content operation.

Note that a control point can attempt to deviate slightly from the restrictions listed in 7.4.1.7.12.2, but the result may be an error. For example, if the upload controller tries to specify multiple <res> elements, then the DMS can choose to fail the request.

7.4.1.7.12.3 If a UPnP AV MediaServer responds with a success to a CDS:CreateObject request for an OCM: upload content operation, then the created CDS item (as returned in the result output argument) must comply with the following rules:

- the @parentID of the created CDS item must match the request's specified ContainerID input argument;
- the <res> element that provides the res@importUri value intended for the content transfer process must comply with the guidelines in 7.4.1.7.19.

M	A	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.12.4 A UPnP AV MediaServer that supports the OCM: upload content operation must be capable of participating in a content transfer process, as described in guideline 7.4.1.7.26.

M	A	DMS	M-DMS	n/a	[52] [55]

These guidelines introduce the second step to the OCM: content upload operation.

7.4.1.7.12.5 A UPnP AV MediaServer control point that supports the OCM: content upload operation must be capable of participating in a content transfer process, as described in guideline 7.4.1.7.26.

M	A	+UP+	M-DMU	MIU	[52] [55]

7.4.1.7.12.6 A UPnP AV MediaServer that creates a new CDS item in an OCM: upload content operation may change metadata values to indicate the support or unsupported nature of OCM operations.

O	C	DMS	M-DMS	n/a	[52] [55]

For example, the DMS may automatically create the @dlna:dlnaManaged attribute or change the @restricted value to indicate support for one or more OCM operations.

7.4.1.7.13 MM/CM: OCM:Create child container operation

7.4.1.7.13.1 If a UPnP AV MediaServer supports the OCM: create child container operation on a CDS container, then it must specify one or more <upnp:createClass> elements to indicate the types of CDS objects that can be created in the container.

M	R	DMS	M-DMS	n/a	[52] [55]

This guideline ensures that control points will know what type of CDS containers can be created in an existing container.

It should be noted that destroying containers is out-of-scope for this version of DLNA guidelines. DLNA assumes that DMS have ownership of their CDS hierarchy, which includes the ability to remove containers through out-of-band mechanisms.

7.4.1.7.13.2 If a UPnP AV MediaServer supports the OCM: create child container operation, then it must allow control points to specify a "DLNA.ORG_AnyContainer" value for the ContainerID input argument in a CDS:CreateObject request.

M	A	DMS	M-DMS	n/a	[52] [55]

The "DLNA.ORG_AnyContainer" allows an upload controller to create a container on a UPnP AV MediaServer without having specific knowledge of where the content should be listed in the CDS hierarchy.

7.4.1.7.13.3 A UPnP AV MediaServer that supports OCM:create child container must also implement support for OCM:upload content.

M	A	DMS	M-DMS	n/a	n/a

7.4.1.7.13.4 If a UPnP AV MediaServer control point is going to start an OCM: create child container operation, then it must invoke CDS:CreateObject with the following rules:

- the ContainerID input argument must indicate "DLNA.ORG_AnyContainer" or a CDS container that supports the OCM: create child container operation;
- the elements input argument must be a CDS container whose <upnp:class> value matches one of the values in the set of <upnp:createClass> elements associated with the CDS container identified by the ContainerID input argument. If specifying "DLNA.ORG_AnyContainer" as the value for the ContainerID input argument, then the value must be object.container or a similarly derived class;
- the elements input argument must include one or more <upnp:createClass> values that describe the types of CDS objects that will be created in the new container. If specifying "DLNA.ORG_AnyContainer" as the value for the ContainerID input argument, then the corresponding <upnp:createClass> (or similarly derived values) must be used.

Table 20 – Required UPnP createClass elements

Media class	Required upnp:class value (for upnp:createClass elements)
Audio	object.item.audioItem
Image	object.item.imageItem
AV	object.item.videoItem

M	A	+UP+	M-DMU	MIU	[52] [55]

If a MediaServer supports the operation, upload controllers or M-DMUs can choose to create new CDS containers that can receive uploaded content in an organized way.

The <upnp:createClass> values that are declared in the CDS:CreateObject request indicate the types of media that the upload controller intends to upload into the new container.

The following conditions may cause the DMS to return an error because specific aspects of the syntax are optional:

- the <upnp:class> value is derived from object.container. DMS may fail this request because object.container is the only value that is mandatory;
- the <upnp:createClass> value is unsupported, even if it is derived from a supported type. DMS may fail this request because only the upnp:createClass values listed in the table are mandatory;
- there is more than one <upnp:createClass> value in the request. DMS may fail this request because only a single <upnp:createClass> is required.

7.4.1.7.13.5 A UPnP AV MediaServer may change a <upnp:createClass> or <upnp:class> value to one of the supported base classes or one of the derived classes if the value is derived from that base class.

O	A	DMS	M-DMS	n/a	[52] [55]

For example, if the upload controller specified a <upnp:createClass> value of object.item.imageItem.photo and a <upnp:class> value of object.container.album, then the DMS may automatically change the values of those elements to object.item.imageItem and object.container, respectively. Similarly, a DMS can change object.container to a derived value, such as object.container.storageFolder.

Creating a container that has <upnp:createClass> values derived from object.container are out of scope.

7.4.1.7.25.3 instructs a DMS to return error code 712 if the specified value is not acceptable to the DMS.

7.4.1.7.13.6 A UPnP AV MediaServer control point creates a new CDS container with intent to follow up with an OCM: upload content operation the new container, then the control point must specify the @dlna:dlnaManaged attribute with bit-0 set to true.

M	A	+UP+	M-DMU	MIU	[52] [55]

In conjunction with the <upnp:createClass> elements, this guideline ensures that a DMS knows the intent of the Upload Controller to upload content to the new container.

7.4.1.7.13.7 A UPnP AV MediaServer that creates a new CDS container in an OCM: create child container operation may change the @restricted and/or @dlna:dlnaManaged metadata values to indicate the support or unsupported nature of OCM operations.

O	C	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.13.8 If a UPnP AV MediaServer receives an OCM: create child container request and one or more of the <upnp:createClass> values specified in the request are unsupported, then the MediaServer must return error code 712 (bad metadata).

M	L	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.13.9 If a UPnP AV MediaServer responds with a success to a CDS:CreateObject request for an OCM: create child container operation, then the created CDS container must have the @dlna:dlnaManaged attribute. Furthermore, the @parentID of the created CDS container (as returned in the result output argument) must match the request's specified ContainerID input argument.

M	A	DMS	M-DMS	n/a	[52] [55]

The @dlna:dlnaManaged attribute is always present in a recursive manner. However, the individual bits on the attribute value may have different true/false values.

7.4.1.7.14 MM/CM: OCM: Destroy item operation

7.4.1.7.14.1 If a UPnP AV MediaServer control point is going to start an OCM: destroy item operation, then it must invoke CDS:DestroyObject with the following rules:

- the ObjectID input argument must indicate a CDS item that supports the OCM: destroy item operation;
- this ObjectID must be for a CDS item that is to be removed from the CDS.

M	R	+UP+	M-DMU	MIU	[52] [55]

The primary usage of this operation is to allow an M-DMU or Upload Controller to remove CDS items from a MediaServer. A control point can make no assumptions about whether any content files will actually be removed.

7.4.1.7.14.2 If a UPnP AV MediaServer responds with a success to a CDS:DestroyObject request for an OCM: destroy item operation, then the DMS must remove the CDS item indicated by the request's ObjectID input argument. A MediaServer that cannot remove the indicated CDS item from the CDS hierarchy must return a SOAP error response.

M	R	DMS	M-DMS	n/a	[52] [55]

DLNA does not specify any mandatory behaviour for whether or not actual content binaries are removed from the local storage of the DMS. The primary expectation of this guideline is that the destroyed CDS item no longer appears in the CDS hierarchy.

7.4.1.7.15 MM/CM – Use of valid values

A UPnP AV MediaServer control point must always specify values for XML attributes and elements in a manner that conforms with the XML attribute's or element's schema.

M	C	+UP+	M-DMU	MIU	[52] [55]

This is a general rule that applies whenever a control point creates or changes a metadata property.

7.4.1.7.16 MM/CM: General use of 7xx error codes

If a UPnP AV MediaServer responds to a CDS:CreateObject request with a UPnP AV error code in the 700 – 799 range, then the UPnP AV MediaServer may use a localized, human-readable error message in the errorDescription tag of the SOAP response.

O	A	DMS	M-DMS	n/a	[52] [55]

The ContentDirectory service specification does not provide adequate granularity for many error scenarios. This guideline allows DMS or M-DMS vendors to provide an error message that can be used by a user for error recovery/and or troubleshooting.

7.4.1.7.17 MM/CM – General use of error code 720

Unless a different error code is mandatory, if a UPnP AV MediaServer receives a CDS action request for a particular upload AnyContainer or any OCM operation that is not supported or invalid, then the UPnP AV MediaServer may return a UPnP error code of 720 (Cannot process the request). The errorDescription tag in the SOAP response may contain a localized, human-readable error message.

O	A	DMS	M-DMS	n/a	[52] [55]

The ContentDirectory service specification does not provide adequate granularity for many error scenarios. As a workaround, DLNA allows implementations to return the error code 720 with a detailed error message. This error message may be used by a consumer for error recovery and/or troubleshooting.

7.4.1.7.18 MM/CM – Invalid 4th field parameters

A UPnP AV MediaServer that receives a CDS:CreateObject that conflicts with the guideline in 7.4.1.3.12.9 may return a UPnP AV error code 712 (bad metadata) or it may return a success response and change the conflicting flag to an appropriate value.

O	C	DMS	M-DMS	n/a	[52] [55]

In content upload usages, the DMS (not the Upload Controller) determines the transport layer options.

7.4.1.7.19 MM/CM – General rule for creating <res> elements that support a content transfer process

7.4.1.7.19.1 If a UPnP AV MediaServer control point invokes CDS:CreateObject and specifies the creation of a <res> element that omits a URI value, then the control point must specify a <res> element that conforms to the following rules:

- the first field of the res@protocolInfo value must be "";

- the second field of the res@protocolInfo value must be "";
- the third field of the res@protocolInfo value must be a valid DLNA mime-type that correlates with the DLNA.ORG_PN value in the fourth field;
- the fourth field of the res@protocolInfo value must have the DLNA.ORG_PN parameter and value. The value must identify the DLNA media format profile of the content binary that will be used in the content transfer process. (It is permissible to include the ci-param.)

The fourth field of the res@protocolInfo value must omit all other 4th field parameters defined by the DLNA guidelines, excluding DLNA.ORG_PN and DLNA.ORG_CI.

The <res> element must omit the res@importUri attribute.

M	C	+UP+	M-DMU	MIU	[52] [55]

A control point that specifies a <res> element without a URI value indicates that it wants to upload content to the MediaServer. A control point can deviate from the restrictions described in this guideline, but the result may be an error. For example, if the control point does not specify a DLNA.ORG_PN parameter, then the MediaServer can return an error.

7.4.1.7.19.2 If a UPnP AV MediaServer creates a <res> element as a result of a CDS:CreateObject and the <res> element specified in the CDS request omits a URI value, then the created <res> element must comply with the following rules:

- the <res> element must omit a URI value;
- the <res> element must have a res@importUri attribute and value. The res@importUri value must indicate a URI that supports a content-transfer process. The length of the URI must be less than or equal to 1024 bytes.

(It is permissible for the MediaServer to ignore or preserve a DLNA.ORG_CI parameter in the 4th field of the res@protocolInfo if the request specifies it.)

M	C	DMS	M-DMS	n/a	[52] [55]

If a MediaServer creates a <res> element intended to receive uploaded content, then it needs to provide a res@importUri attribute and value. This URI tells an Upload Controller which URI to perform an HTTP POST operation.

Note that the value of the <res> element needs to be empty until the content is actually available for serving as described in 7.4.1.7.27.4.

7.4.1.7.19.3 If a UPnP AV MediaServer control point invokes CDS:CreateObject and specifies the creation of a <res> element without a URI value and the content is profiled as MPEG_PS_NTSC or MPEG_PS_PAL and it has discontinuous SCR and/or PTS, then the MediaServer control point must specify a res@dlna:ifoFileURI attribute with an empty value.

The prefix for res@dlna:ifoFileURI must be "dlna:" and the namespace must be "urn:schemas-dlna-org:metadata-1-0/".

M	A	+UP+	M-DMU	MIU	[52] [55]

Upload Controllers or M-DMUs may need to upload an IFO file. Providing an empty res@dlna:ifoFileURI signals the MediaServer to provide a URI value for it. In many cases, the MediaServer will include the res@dlna:importIfoFileURI attribute in the CDS:CreateObject response to indicate that the MediaServer will receive the IFO file. Upload Controllers and M-DMUs then perform a content transfer process to this URI in the same manner as a res@importUri.

7.4.1.7.19.4 If a UPnP AV MediaServer creates a <res> element in response to a CDS:CreateObject that has a res@dlna:ifoFileURI attribute with an empty value, then the UPnP AV MediaServer must do one of the following:

- preserve the res@dlna:ifoFileURI attribute with an empty value and add the res@dlna:importIfoFileURI attribute with a URI value that supports the content transfer process for the IFO file associated with the content binary;
- preserve the res@dlna:ifoFileURI attribute with an empty value and omit the res@dlna:importIfoFileURI attribute to indicate that the MediaServer will automatically create an IFO file (and expose it through the res@dlna:ifoFileURI) after a successful content transfer process on the res@importUri;
- omit the res@dlna:ifoFileURI attribute and omit the res@dlna:importIfoFileURI attribute to indicate that the MediaServer will automatically generate an equivalent content binary without discontinuities after a successful content transfer process on the res@importUri.

M	A	DMS	M-DMS	n/a	[52] [55]

A MediaServer that is able to create a <res> element for such content (that is known to have discontinuous SCR and/or PTS) must provide a mechanism for receiving the associated IFO file or must generate the IFO file automatically after receiving the content binary.

Note that the res@dlna:ifoFileURI attribute remains an empty value until the MediaServer is ready to serve the IFO file, such as described in 7.4.1.7.27.7.

7.4.1.7.19.5 If a UPnP AV MediaServer provides res@dlna:importIfoFileURI and an Upload Controller has intention to transmit a MPEG_PS_NTSC/PAL content and an associated IFO file to the MediaServer, then the upload controller must completely transmit the IFO file before sending the MPEG_PS_NTSC/PAL content.

M	L	+UP+	M-DMU	n/a	[52] [55]

In the typical case, MediaServer uses the IFO file before receiving the content data to check its data size and boundary information in it.

7.4.1.7.19.6 If a UPnP AV MediaServer control point invokes CDS:CreateObject and specifies the creation of a <res> element without a URI value, then the MediaServer control point should specify res@size attribute with a value equal to the byte length or equal to an estimated byte length of the content that will be sent during the content transfer process.

S	A	+UP+	M-DMU	MIU	[52] [55]

This allows a DMS or M-DMS to know how large the uploaded content will be and gives it the opportunity to reserve local storage space for the content.

Please note that the size of the content binary that is sent during the content transfer process can be different from the res@size value. For example, an upload controller may upload a content binary that is actually a conversion from another media format. In such a scenario, it may be very difficult to anticipate the exact size, so the upload controller specifies a res@size value that is a bit larger than the estimated size. During the content transfer process, the DMS will be able to determine the correct size of the content.

7.4.1.7.19.7 If a UPnP AV MediaServer successfully completes a content transfer process for the <res> element (and if res@size is present), then the MediaServer must correct the res@size value to match the length of the content binary, if the indicated value does not match the byte length of the uploaded content binary.

M	A	DMS	M-DMS	n/a	[52] [55]

Once content has been uploaded and the DMS or M-DMS can serve the content to other endpoints, the res@size value needs to be accurate. Estimated values will cause problems for rendering endpoints that rely on that metadata.

7.4.1.7.20 MM/CM – General rule for creating <res> elements that support a resume content transfer process

7.4.1.7.20.1 A UPnP AV MediaServer control point may support the resume content transfer operation. MediaServer control points may also make multiple resume content transfer operations.

The resume content transfer operation is a retry attempt for a failed content transfer process such that the content transfer starts from the point of transfer failure from a previous attempt.

O	A	+UP+	M-DMU	MIU	[52] [55]

Upload Controllers and M-DMUs are not required to support the ability to retry or resume a failed content transfer.

The DLNA guidelines provide interoperability rules for resume content transfer. The DLNA guidelines do not provide any normative mechanism for a retry attempt such that the content transfer process retries from the beginning of the content binary.

7.4.1.7.20.2 If resume content transfer is supported, then retry IFO attempt must also be supported.

For MediaServers, this means that returning a CDS:CreateObject response that has res@dlna:resumeUpload="1" must mean that resume content transfer and retry IFO attempt are supported.

For MediaServer control point, this means that issuing a CDS:CreateObject request with res@dlna:resumeUpload="1" must mean the MediaServer control point is capable of using resume content transfer and retry IFO attempt if an error occurs during the transmission of the content binary or IFO file.

A retry IFO attempt is defined as a retry attempt for a failed content transfer process attempt for an IFO file, such that the transfer begins from byte index 0 of the IFO file and the correspond HTTP POST request omits the CONTENT-Range HTTP header.

M	A	DMS +UP+	M-DMS M-DMU	MIU	[52] [55]

7.4.1.7.20.3 If a UPnP AV MediaServer control point invokes CDS:CreateObject and wants to create a <res> element that supports the resume content transfer operation, then it must specify a res@dlna:resumeUpload attribute with "1" value in a call to CDS:CreateObject.

The prefix for res@dlna:resumeUpload must be "dlna:" and the namespace must be "urn:schemas-dlna-org:metadata-1-0/".

M	A	+UP+	M-DMU	MIU	[52] [55]

Upload controllers create <res> elements with a `res@dlna:resumeUpload="1"` to determine if the MediaServer supports the resume content transfer operation. Upload controllers must not specify `res@dlna:resumeUpload="1"` unless they intend to recover from a failed content transfer process by using the CONTENT-Range HTTP header in an HTTP POST request (i.e., using resume content transfer). The only other normative recovery mechanism for failed content uploads is to restart from a new upload AnyContainer or OCM: upload content operation.

7.4.1.7.20.4 If a UPnP AV MediaServer creates a new <res> element as a result of a CDS:CreateObject request that specified the `res@dlna:resumeUpload` attribute with a value of "1" and if the MediaServer supports the resume content transfer operation for the created item, then the MediaServer must preserve the value "1" for the `res@dlna:resumeUpload` attribute.

M	A	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.20.5 If the MediaServer creates the <res> element, it will confirm the support of the resume content transfer operation by preserving the `res@dlna:resumeUpload="1"` attribute and value.

A MediaServer respond `res@dlna:resumeUpload="1"` should keep 30 min the object when upload has failed.

If a UPnP AV MediaServer receives a CDS:CreateObject that specifies `res@dlna:resumeUpload="0"` or omits `res@dlna:resumeUpload`, then the MediaServer's CDS:CreateObject response must specify `res@dlna:resumeUpload="0"` or omit `res@dlna:resumeUpload`.

M	A	DMS	M-DMS	n/a	[52] [55]

MediaServers must never enable resume content transfer unless the upload controller requests that the feature apply to the upload operation because a control point can intentionally choose to rely on the required Auto-Destroy behaviour when resume content transfer is not supported.

See 7.4.1.7.28.2 for more information on MediaServer behaviour when resume content transfer is not supported.

7.4.1.7.20.6 If a UPnP AV MediaServer creates a new <res> element as a result of a CDS:CreateObject request that specified the `res@dlna:resumeUpload` attribute with a value of "1" and if the MediaServer cannot support the resume content transfer operation for the created item, then the MediaServer must do one of the following:

- set the `res@dlna:resumeUpload` attribute to "0" in the created <res> element;
- omit the `res@dlna:resumeUpload` attribute in the created <res> element

M	A	DMS	M-DMS	n/a	[52] [55]

DMS implementations that do not support the resume content transfer operation must specify a value of "0" or omit the `res@dlna:resumeUpload` attribute to inform upload controllers that the operation is not supported.

7.4.1.7.20.7 If a UPnP AV MediaServer control point attempts to recover from a failed content transfer process and both of the following conditions are true, then the Upload Controller must use the resume content transfer operation as the initial recovery attempt.

- MediaServer control point specified `res@dlna:resumeUpload="1"` in a CDS request

- MediaServer preserved `res@dlna:resumeUpload="1"` in the CDS response that created the `<res>` element

M	A	+UP+	M-DMU	MIU	[52] [55]

An M-DMU or upload controller that specifies `res@dlna:resumeUpload="1"` must use an HTTP POST request with the CONTENT-Range HTTP header whenever possible. As a fallback, an upload controller is permitted to restart the upload process by restarting with an OCM: upload content operation. Some example scenarios using the fallback process are listed here:

- the CDS object or `<res>` element no longer exists on the DMS (for example, the suspend period is too long);
- the resume content transfer attempt has failed for some reason. In this case, the upload controller should use an OCM: destroy item operation before restarting with an OCM: upload content operation.

7.4.1.7.20.8 If a UPnP AV MediaServer control point attempts to recover from a failed content transfer process and at least one of the conditions are true, then upload controller may attempt the recovery by starting over with an upload AnyContainer or OCM: upload content operation and follow up with a new content transfer process:

- MediaServer control point did not specify `res@dlna:resumeUpload="1"` in the CDS request that created the `<res>` element;
- MediaServer control point received a CDS response with `res@dlna:resumeUpload=0` or response omits `res@dlna:resumeUpload` attribute from the created `<res>` element.

O	A	+UP+	M-DMU	MIU	[52] [55]

7.4.1.7.21 MM `res@dlna:uploadedSize`

7.4.1.7.21.1 If a UPnP AV MediaServer supports the resume content transfer operation for a `<res>` element, then the `<res>` element must have a `res@dlna:uploadedSize` attribute.

The value of `res@dlna:uploadedSize` is the byte length of the content binary (not including the size of an uploaded IFO file) that was received during a content transfer process that failed.

The prefix for `res@dlna:uploadedSize` must be "dlna" and the namespace must be "urn:schemas-dlna-org:metadata-1-0/".

M	A	DMS	M-DMS	n/a	[52] [55]

An M-DMU or Upload Controller can perform a resume content transfer by using an HTTP POST request with the CONTENT-Range header.

The `res@dlna:uploadedSize` does not represent the number of bytes that have been uploaded for an IFO file.

7.4.1.7.21.2 If a UPnP AV MediaServer does not support the resume content transfer operation for a `<res>` element, then it may omit the `res@dlna:uploadedSize` attribute.

O	A	DMS	M-DMS	n/a	[52] [55]

Usually the `res@dlna:uploadedSize` is omitted but it is not mandatory because it can simplify the MediaServer implementation. There is no normative use for this property in such scenarios.

7.4.1.7.22 MM/CM – General rules for <res> elements

7.4.1.7.22.1 A UPnP AV MediaServer must have at least one of the following URI values for each <res> element.

- URI value for the <res> element
- URI value for the res@importUri attribute

M	L	DMS	M-DMS	n/a	[52] [55]

A <res> element without one of these URI values is incomplete. If both URI values are present, then the DMS indicates that uploading endpoints can overwrite the binary for a specific <res> element.

7.4.1.7.22.2 If a UPnP AV MediaServer creates a <res> element in response to a CDS:CreateObject, then the created <res> element may change the set of metadata attributes that were specified in the request.

O	L	DMS	M-DMS	n/a	[52] [55]

This guideline allows a control point to create <res> elements without knowledge of attributes supported by the MediaServer.

For example, a control point can specify a new <res> element with the res@size attribute but the created <res> element may omit res@size because the attribute is not supported.

Another example is when the DMS creates a <res> element that omits the res@dlina:ifoFileURI attribute when the request originally specified one. In this situation, the DMS created the <res> element because the DMS will create and serve an IFO file after it receives the movie file or the DMS will modify the movie file so that it does not have any discontinuities.

7.4.1.7.23 MM/CM: General rules for CDS – CreateObject request syntax

7.4.1.7.23.1 If a UPnP AV MediaServer control point invokes CDS:CreateObject, then it must apply the mandatory metadata encoding rules for the following guidelines in the elements input argument.

7.4.1.3.1 MM UPnP AV control point tolerance of unknown property

7.4.1.3.2 MM DIDL-Lite

7.4.1.3.3 MM DIDL-Lite max metadata

7.4.1.3.5 MM DIDL-Lite boolean values

7.4.1.3.6 MM upnp – Class

7.4.1.3.7 MM DIDL-Lite dc:date

7.4.1.3.9 MM DIDL-Lite desc element use

7.4.1.3.10 MM URI

7.4.1.3.11 MM DIDL-Lite recommended metadata

7.4.1.3.12 MM protocolInfo

7.4.1.3.15 MM DIDL-Lite protocolInfo values

7.4.1.3.16 MM protocolInfo values – 4th field

7.4.1.3.17 MM pn-param (DLNA.ORG_PN Parameter)

7.4.1.3.22 MM ci-param (conversion indicator flag)

M	L	+UP+	M-DMU	MIU	[52] [55]

Control points are required to honor basic metadata restrictions that govern CDS metadata.

7.4.1.7.23.2 If a UPnP AV MediaServer control point invokes CDS:CreateObject, then it must apply the rules specified in 7.4.1.3.4.2 and 7.4.1.3.4.3 (in MM DIDL-Lite non-empty metadata) to all metadata, except the <res> element and @id attribute.

M	L	+UP+	M-DMU	MIU	[52] [55]

The <res> element in a CDS:CreateObject is exempt from the non-empty value conventions because Upload Controllers do not specify the <res> URI value for some tasks, such as the upload AnyContainer or OCM: upload content operations.

7.4.1.7.23.3 If a UPnP AV MediaServer control point invokes CDS:CreateObject, then it must adhere to the following rules when specifying the Elements input argument:

- the XML must exclude the <?xml> declarator and use UTF-8 encoding;
- the <DIDL-Lite> element must be the top-most element;
- the <DIDL-Lite> element must have a single <item> or a single <container> child element;
- at minimum, the <item> or <container> element must specify the @id, @parentID, @restricted, dc:title, and upnp:class metadata properties;
- the @id value must be an empty string ("");
- the @parentID value must match the ContainerID input argument;
- the @restricted value must be "0";
- the dc:title value must comply with all DLNA-specified metadata restrictions.

Example request:

- CDS:CreateObject("10", "<DIDL-Lite xmlns:dc='http://purl.org/dc/elements/1.1/' xmlns:upnp='urn:schemas-upnp-org:metadata-1-0/upnp/' xmlns='urn:schemas-upnp-org:metadata-1-0/DIDL-Lite'> <item id='' restricted='0' parentID='10'> <dc:title>A picture in the park</dc:title> <upnp:class>object.item.imageItem</upnp:class> <res protocolInfo='*:image/jpeg:DLNA.ORG_PN=JPG_LRG'></res> </item> </DIDL-Lite>")

M	L	+UP+	M-DMU	MIU	[52] [55]

This guideline specifies the baseline requirements for the metadata specified in a CDS:CreateObject request.

Upload controllers must be aware that the created object can have metadata that is slightly different from what is sent. For example,

the @id value of the new object will be set;

the @parentID of the new object will be set (in the case where DLNA.ORG_AnyContainer is used);

informative metadata, such as dc:title and dc:creator, may be truncated;

the @restricted value may be changed;

the returned object can include @dlna:dlnaManaged attribute;

upnp:class or upnp:createClass values can be changed to a derived or a base class value for audio, image, audio/video, and container objects.

7.4.1.7.23.4 If a UPnP AV MediaServer control point invokes CDS:CreateObject, then it should provide all metadata defined in 7.4.1.3.11 for the given upnp:class.

S	C	+UP+	M-DMU	MIU	[52] [55]

Control points should provide recommended metadata because it is useful for a user. If the MediaServer does not support the specified metadata, the CDS:CreateObject response will omit the unsupported metadata, as described in the guideline.

7.4.1.7.23.5 If a UPnP AV MediaServer control point invokes CDS:CreateObject, then it may specify additional metadata properties in a call to CDS:CreateObject, than those specified in 7.4.1.7.23.3.

O	R	+UP+	M-DMU	MIU	[52] [55]

Control points are allowed to specify more than just the minimal metadata.

Note that control points are expected to handle a possible outcome described in 7.4.1.7.24.3.

7.4.1.7.23.6 If a UPnP AV MediaServer control point invokes CDS:CreateObject with optional metadata, then the control point must ensure that all necessary namespaces are properly declared.

M	R	+UP+	M-DMU	MIU	[52] [55]

7.4.1.7.23.7 A UPnP AV MediaServer that allows the creation of <dc:date> elements must allow the control point to specify any valid form of <dc:date>, as defined by guideline 7.4.1.3.7.

M	C	DMS	M-DMS	n/a	[52] [55]

This guideline allows the control point to specify any valid form for <dc:date>. Control points are not to assume that the format they specified in the request will be the final form of the <dc:date>.

7.4.1.7.23.8 A UPnP AV MediaServer that allows the creation of <dc:date> elements may change the form of the <dc:date> value to match any valid form, as defined by guideline 7.4.1.3.7.

O	C	DMS	M-DMS	n/a	[52] [55]

This guideline allows MediaServers the flexibility of using a consistent <dc:date> form for all of their CDS objects.

7.4.1.7.23.9 If a UPnP AV MediaServer control point invokes CDS:CreateObject and the created object needs to support one or more OCM operations, then the MediaServer control point should specify the appropriate value for the @dlna:dlnaManaged attribute in the request.

S	A	+UP+	M-DMU	MIU	[52] [55]

For example, if an upload controller requires a created container to support the OCM: upload content operation, then the control point needs to specify a CDS container that supports the OCM: upload content operation in the CDS:CreateObject request by specifying the @dlna:dlnaManaged attribute with the appropriate bits set.

7.4.1.7.24 MM/CM – General rules for CDS:CreateObject response syntax

7.4.1.7.24.1 If a UPnP AV MediaServer that implements CDS:CreateObject, then it must be able to support a request that specifies the properties specified in 7.4.1.7.23.3.

Example response:

- CDS:CreateObject("12","<DIDL-Lite xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/" xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/"> <item id="12" parentId="10" restricted="0"> <dc:title>A picture in the park</dc:title> <res importUri="http://192.168.1.1/item?id=12" protocolInfo="*:image/jpeg:DLNA.ORG_PN=JPEG_LRG"/> <upnp:class>object.item.imageItem</upnp:class> <upnp:album>Photo Folder1</upnp:album> </item> </DIDL-Lite="")

M	C	DMS	M-DMS	n/a	[52] [55]

It should be noted that a UPnP AV MediaServer is not required to support the creation of container objects. In such cases, the proper behaviour is to return UPnP error code 712, as described in 7.4.1.7.23.2.

7.4.1.7.24.2 If a UPnP AV MediaServer implements CDS:CreateObject, then it should accept the creation of recommended properties corresponding to the specified upnp:class, which are defined in the guideline 7.4.1.3.11.

S	C	DMS	M-DMS	n/a	[52] [55]

Although not required, metadata such as dc:creator and upnp:album can be useful to a user.

7.4.1.7.24.3 If a UPnP AV MediaServer creates a new CDS object in response to a CDS:CreateObject request (that conforms to DLNA guidelines) and the request specifies metadata properties that are unsupported by the MediaServer, then the MediaServer must return a success response with the Result output that includes the metadata supported by the MediaServer.

Furthermore, if the MediaServer is able to return a success for a CDS:CreateObject request that specifies only the baseline metadata properties, then it must also return a success for a CDS:CreateObject request that specifies the same baseline metadata properties and values with additional optional metadata properties.

M	L	DMS	M-DMS	n/a	[52] [55]

The general approach for CDS:CreateObject is that a DMS must tolerate the presence of optional metadata properties (XML elements and attributes). The tolerance requirement also extends to 4th field parameters of protocolInfo, as required by 7.4.1.3.26.3.

The general exception to this approach is that a DMS is not required to tolerate <upnp:class> or <upnp:createClass> values that are unacceptable to a DMS. The primary reason for this exception is that derived classes can often imply a semantic difference. For example object.item.audioItem.audioBroadcast is not equivalent to an object.item.audioItem because the former implies that the stream is live. It is permissible for a DMS to tolerate a diversity of <upnp:createClass> and <upnp:class> values by automatically changing the specified values, but such behaviour is not required.

A UPnP AV MediaServer that returns unsupported metadata in the Result but responds to CDS:Search and CDS:Browse without that metadata is behaving inconsistently. Returning with an error makes it difficult for control points to provide an information-rich experience. For this reason, DLNA requires that the CDS:CreateObject response includes only the metadata supported by the MediaServer. In this case, metadata supported by the MediaServer has the following characteristics:

- must minimally include the MediaServer's supported metadata properties (XML elements and attributes) that was specified in the request;
- and may optionally include additional metadata properties (XML elements and attributes) that the MediaServer added as part of the success CDS:CreateObject response.

7.4.1.7.24.4 A UPnP AV MediaServer that creates a new CDS object, in response to a CDS:CreateObject request, may add additional metadata properties (XML elements or attributes) in the CDS:CreateObject response.

O	C	DMS	M-DMS	n/a	[52] [55]

The following are some examples of when a DMS might add additional metadata properties.

- The DMS adds the @dlna:dlnaManaged attribute to indicate the supported OCM operations for a created CDS object.
- The DMS adds one or more upnp:createClass elements (in addition to those specified in the CDS:CreateObject request) to indicate additional objects that can be created in a new CDS container.

7.4.1.7.24.5 A UPnP AV MediaServer that receives a CDS:CreateObject request may change the values of the @restricted and/or @dlna:dlnaManaged so long as the new values indicate a superset of the requested OCM operation.

O	A	DMS	M-DMS	n/a	[52] [55]

DMS or M-DMS devices are always allowed to increase the set of supported operations on a created object. Otherwise, the DMS is required to return an error when the requested OCM operations cannot be provided for the CDS object described in the CDS:CreateObject request.

7.4.1.7.24.6 If a UPnP AV MediaServer that receives a CDS:CreateObject request that specifies the creation of a CDS object with support for one or more OCM operations that cannot be supported for the specified object, then the MediaServer must return a failure with UPnP AV error code 712 (bad metadata).

M	A	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.25 MM/CM – General rules for CDS:CreateObject errors

7.4.1.7.25.1 If a UPnP AV MediaServer receives metadata values that are invalid in a CDS:CreateObject request, then the MediaServer must return UPnP AV error code 712 (bad metadata).

Invalid values are values that are not valid for the data type (as defined by the appropriate schema) or values that are not supported by the MediaServer implementation.

Valid, non-empty values are values that are valid for the data type specified for the metadata property and also not composed entirely of white space characters.

M	L	DMS	M-DMS	n/a	[52] [55]

This ensures that DLNA-specified restrictions for metadata are enforced.

It should be noted that a UPnP AV MediaServer that removes the entire metadata property from the result output is informing the control point that the specified property is not supported, rather than informing the control point that the value is invalid. This is described in 7.4.1.7.24.3.

Although this guideline applies to the MediaServer, control points are expected to provide valid metadata values, as required by 7.4.1.7.15 and 7.4.1.7.24.

7.4.1.7.25.2 If a UPnP AV MediaServer returns a CDS:CreateObject response with a UPnP error code, it must not result in the creation of a new CDS object.

M	L	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.25.3 If the UPnP AV MediaServer does not support the CDS:CreateObject action because the specified upnp:class or res@protocolInfo is not acceptable to the MediaServer, it must respond with an error code 712 (Bad Metadata).

M	L	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.25.4 If a UPnP AV MediaServer receives a CDS:CreateObject request with zero or multiple <res> elements, then it may return a UPnP AV error code of 712 (bad metadata).

O	C	DMS	M-DMS	n/a	[52] [55]

MediaServers are not required to support zero or multiple <res> elements. This behaviour is not mandatory because a UPnP AV MediaServer may be able to support the creation of a container object with zero <res> elements.

7.4.1.7.25.5 If a UPnP AV MediaServer receives a CDS:CreateObject request where the <res> element specifies a URI value or a res@importUri value, then it may return a UPnP AV error code of 712 (bad metadata).

O	C	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.25.6 If a UPnP AV MediaServer cannot accept a CDS:CreateObject request due to the processing capacity or current state of the device, then it must respond with error code 720 (cannot process the request).

M	L	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.25.7 If a UPnP AV MediaServer cannot accept a CDS:CreateObject request due to the lack of storage capacity, then the MediaServer must return a UPnP error response with error code of 720 (cannot process the request).

M	L	DMS	M-DMS	n/a	[52] [55]

A MediaServer can return this error code as a result of a res@size value specified in a CDS:CreateObject request. A DMS can also return this value when the upload capacity for the DMS has been reached.

7.4.1.7.26 MM/CM – Content transfer process

7.4.1.7.26.1 If a UPnP AV MediaServer control point is going to initiate a content transfer process for the first file (either the IFO file or the actual content binary) of a CDS object, then it must do so within 30 s of the last event described below.

- The CDS:CreateObject response from a Upload AnyContainer or an OCM: upload content operation.

If a UPnP AV MediaServer control point uploads an IFO file, then the IFO file must be uploaded first and the transfer of the actual content binary must be started within 30 s of completing the IFO file transfer.

M	L	+UP+	M-DMU	MIU	[52] [55]

Within 30 s of creating the CDS object, the upload controller begins its first content transfer for the CDS object.

7.4.1.7.26.2 A UPnP AV MediaServer control point must utilize the URI provided in a res@importUri or a res@dlina:importIfoFileURI as the destination URI for a content transfer process.

UPnP AV MediaServer control points must only upload content binaries to the res@importUri.

UPnP AV MediaServer control points must only upload IFO files to the URI value of res@dlina:importIfoFileURI.

M	L	+UP+	M-DMU	MIU	[52] [55]

The CDS specification is ambiguous as to whether a <res> element's URI value can be used to overwrite content. These guidelines limit the control to using import URI values for both initial and overwriting content transfers. See 7.5.5.6.1 for HTTP client implications.

IFO files are not considered content binaries, so uploading an IFO file to a res@importUri file is expressly prohibited.

For additional information on performing a content transfer process for either a content binary or an IFO file; see 7.5.5.6.1.

For additional information on creating a <res> element for uploading IFO files, see 7.4.1.7.19.3.

7.4.1.7.26.3 If a UPnP AV MediaServer control point fails to completely upload an IFO file and if res@dlna:resumeUpload="1" and if the control point initiates a retry IFO attempt, then the control point must initiate the retry IFO attempt within 30 min of the failure.

M	A	+UP+	M-DMU	MIU	[52] [55]

Retry attempts to transfer the IFO file must start within 30 min of the time of failure because the MediaServer will destroy the CDS object after 35 min has elapsed since the failure.

7.4.1.7.26.4 If a UPnP AV MediaServer control point fails to completely upload a content binary and if res@dlna:resumeUpload="1" and if the control point initiates a resume content transfer, then the control point must initiate the resume content transfer within 30 min of the failure.

M	A	+UP+	M-DMU	MIU	[52] [55]

Resume attempts to transfer the content binary file must start within 30 min of the time of failure because the MediaServer will destroy the CDS object after 35 min has elapsed since the failure.

7.4.1.7.26.5 A UPnP AV MediaServer control point must not attempt to use retry IFO transfer or resume content transfer unless res@dlna:resumeUpload="1".

M	A	+UP+	M-DMU	MIU	[52] [55]

7.4.1.7.26.6 A UPnP AV MediaServer must facilitate a content transfer process by supporting an HTTP POST request on a URI for a res@importUri or a res@dlna:importIfoFileURI.

M	C	DMS	M-DMS	n/a	[52] [55]

DLNA defines HTTP POST as the only mechanism for a content transfer process. DLNA may define additional mechanisms for a content transfer process in the future.

7.4.1.7.27 MM/CM – General rules after a successful content transfer process

7.4.1.7.27.1 If a UPnP AV MediaServer successfully completes a content transfer process, then the MediaServer may keep or remove the `res@importUri` attribute.

O	C	DMS	M-DMS	n/a	[52] [55]

A MediaServer that keeps the `res@importUri` attribute is informing the network that an upload controller can overwrite the content that was uploaded.

7.4.1.7.27.2 If a UPnP AV MediaServer keeps the `res@importUri` attribute and creates a URI value for the `<res>` element (to signal that the content source will serve the URI), then the MediaServer may specify the same URI value for both the `res@importUri` and the URI value of a `<res>` element.

O	C	DMS	M-DMS	n/a	[52] [55]

MediaServers are permitted to use the same URI for both the `<res>` URI value and the `res@importUri` value. In such cases, MediaServers need to ensure that the `<res>` value is not exposed until after the content has been received, as described in 7.4.1.7.27.4.

7.4.1.7.27.3 If a UPnP AV MediaServer keeps the `res@dlna:ifoFileURI` attribute and creates a `res@dlna:importIfoFileURI` value for the `<res>` element (to signal that the Content Source will serve the URI), then the MediaServer may specify the same URI value for both the `res@dlna:importIfoFileURI` and the `res@dlna:ifoFileURI` value of a `<res>` element.

O	C	DMS	M-DMS	n/a	[52] [55]

MediaServers are permitted to use the same URI for both the `<res>` URI value and the `res@importUri` value. In such cases, MediaServers need to ensure that the `<res>` value is not exposed until after the content has been received, as described in 7.4.1.7.27.4.

7.4.1.7.27.4 If a UPnP AV MediaServer is going to serve uploaded content to other endpoints, then the DMS must provide the URI value of the `<res>` element only after the DMS can serve the content.

Example of `<res>` before content is uploaded to the DMS:

- `<res protocolInfo="http-get:*:image/jpeg*:DLNA.ORG_PN=JPEG_LRG" importUri="http://192.168.1.10/upload?file=content-data-1234.jpg"/>`
- example of `<res>` after content is uploaded to the DMS;
- `<res protocolInfo="http-get:*:image/jpeg*:DLNA.ORG_PN=JPEG_LRG"> http://192.168.1.10/content-data-1243.jpg</res>`.

M	L	DMS	M-DMS	n/a	[52] [55]

A UPnP AV MediaServer that advertises a `<res>` URI value before the content can actually be served is making a false claim that the content is available.

It should be note that a DMS that serves partially uploaded content is responsible for conforming to the guidelines related to the media format profile.

7.4.1.7.27.5 A UPnP AV MediaServer may automatically create additional <res> elements (with network-accessible URI values and protocolInfo values) after a successful content transfer process.

O	C	DMS	M-DMS	n/a	[52] [55]

MediaServers may want to provide additional <res> elements to support content transformations or additional transport protocols.

7.4.1.7.27.6 If a UPnP AV MediaServer is going to serve uploaded content that has an IFO file, then the MediaServer must provide the URI value of the <res> element after the content source can provide the IFO file.

M	C	DMS	M-DMS	n/a	[52] [55]

Advertising content with SCR/PTS discontinuities without providing an IFO file is a violation of 7.4.1.4.8.2.

7.4.1.7.27.7 If a UPnP AV MediaServer is going to serve uploaded content that has an IFO file, and successfully receives an IFO file in a content transfer process, then the MediaServer must provide a res@dlna:ifoFileURI attribute with a URI value for the IFO file. Until the MediaServer has the IFO file, the res@dlna:ifoFileURI must have an empty value.

Example timeline (with additional comments for other hypothetical situations after ***)

- (1) Upload controller or M-DMU invokes CDS:CreateObject with a <res> element for MPEG_PS_NTSC. The request includes res@dlna:ifoFileURI="" to indicate that it will also upload an IFO file. The request also includes res@dlna:resumeUpload="1" to indicate that the upload controller will use resume features if a failure occurs during a transfer. *** If the upload controller/M-DMU is not uploading an IFO file, it would have omitted res@dlna:ifoFileURI. If the upload controller/M-DMU was not going to use resume, it would have omitted res@dlna:resumeUpload.
- (2) The DMS or M-DMS responds with success. The response includes a res@importUri and res@dlna:importIfoFileURI, with URI values that will receive the uploaded content. The response preserves the res@dlna:resumeUpload="1" to indicate resume is supported. *** If the DMS/M-DMS does not support resume, it would have omitted res@dlna:resumeUpload or set the value to "0". A DMS/M-DMS never returns res@dlna:resumeUpload="1" unless requested to do so.
- (3) The upload controller begins uploading the IFO file.
- (4) During the transfer from [3], an error occurs and the transfer fails.
- (5) Within 30 min of the error in [4], the upload controller or M-DMU issues an HTTP POST to retry the transfer of the IFO file. The upload controller/M-DMU does not use CONTENT-Range when retrying an IFO transfer. *** If the upload controller/M-DMU does not do the retry, then the DMS/M-DMS would automatically destroy the new CDS object (from [1]). The DMS/M-DMS waits at least 35 min from the failure time in [4] before automatically destroying. In scenarios where resume is not enabled, the DMS/M-DMS would destroy the CDS object (from [1]) at any time after the error in [4] occurred.
- (6) After completing the retry IFO transfer from [5], the upload controller/M-DMU initiates the transfer of the content binary within 30 s. *** If the upload controller/M-DMU is not uploading an IFO, then it would have started the content transfer within 30 s of receiving the CDS:CreateObject response.
- (7) During the transfer from IEEE 802.1D, an error occurs and the transfer fails.

- (8) Within 30 min of the error in IEEE 802.1Q, the upload controller or M-DMU issues an HTTP POST with the CONTENT-Range header to resume the transfer from the byte position where the failure occurred. *** If the upload controller/M-DMU does not do the resume, then the DMS/M-DMS would automatically destroy the new CDS object (from [1]). The DMS/M-DMS waits at least 35 min from the failure time in IEEE 802.1Q before automatically destroying. In scenarios where resume is not enabled, the DMS/M-DMS would destroy the CDS object (from [1]) at any time after the error in IEEE 802.1Q occurred.
- (9) The content transfer from IEEE 802.1D succeeds and the DMS/M-DMS advertises URI values for the <res> element and the res@ifoFileURI.
- (10) The DMS or M-DMS also automatically creates a new <res> element for its locally converted version of MPEG_PS_NTSC_XAC3

M	A	DMS	M-DMS	n/a	[52] [55]

Just as a MediaServer uses the <res> URI value to indicate that the content is available for content receivers, the MediaServer also uses the res@dlina:ifoFileURI attribute to indicate that the IFO file is available for content receivers.

7.4.1.7.28 MM/CM – Auto-Destroy behaviour for a failed or partial content transfer process

7.4.1.7.28.1 If a UPnP AV MediaServer implements CDS:CreateObject for use with the listed operations, then it must implement Auto-Destroy behaviour.

Listed operations:

- upload AnyContainer;
- OCM: upload content.

The Auto-Destroy behaviour must be defined as removing the created CDS object from the CDS hierarchy exposed by the MediaServer, such that the CDS object cannot be found when browsing or searching the MediaServer.

The time when the CDS object is actually removed from the CDS hierarchy is not entirely defined by the guidelines. However, other guidelines in 7.4.1.7.28 have restrictions when the CDS object is automatically removed.

All guidelines in 7.4.1.7.28 assume ideal network conditions and interoperability is not guaranteed in scenarios involving user-initiated or out-of-scope system events.

M	C	DMS	M-DMS	n/a	[52] [55]

A DMS that supports the upload system usage needs to automatically destroy CDS objects when the files are not properly received by the MediaServer.

The guidelines guarantee interoperability only under ideal network conditions. Scenarios that involve user-initiated events or out-of-scope system events do not guarantee interoperability. Examples include

- another control point invoking CDS:DestroyObject causes a failure during an upload AnyContainer operation;
- the MediaServer product cancelling all active uploads because it is starting to record a video broadcast to local storage.

The actual time when the MediaServer destroys such CDS items is not defined by DLNA. Some examples include a 35 min timer mechanism, a clean-up mechanism that executes when the device has been idle for an extended period of time, or a cleanup operation that only executes when certain CDS actions are called.

7.4.1.7.28.2 If `res@dlna:resumeUpload="0"` or `res@dlna:resumeUpload` is omitted, a UPnP AV MediaServer must perform Auto-Destroy behaviour on CDS objects (created through upload AnyContainer or OCM: upload content) under these conditions:

- 35 s have elapsed since the CDS:CreateObject response was completely sent and the first content transfer process (for either the IFO file or the content binary) has not started;
- 35 s have elapsed since the CDS object's IFO was completely received and the content transfer process for the content binary has not started;
- the content transfer process of the content binary or the IFO file has failed.

M	C	DMS	M-DMS	n/a	[52] [55]

This guideline explains when a MediaServer needs to automatically destroy CDS objects that were created from upload AnyContainer or OCM: upload content.

The guideline explicitly covers the case where resume content transfer, as described in 7.4.1.7.20, is not supported.

7.4.1.7.28.3 If `res@dlna:resumeUpload="1"`, a UPnP AV MediaServer must perform Auto-Destroy behaviour on CDS objects (created through upload AnyContainer or OCM: upload content) under these conditions:

- 35 s have elapsed since the CDS:CreateObject response was completely sent and the first content transfer process (for either the IFO file or the content binary) has not started;
- 35 s have elapsed since the CDS object's IFO was completely received and the content transfer process for the content binary has not started;
- 35 min have elapsed since the failure of a content transfer process for the CDS object's IFO and retry IFO attempt has not started;
- 35 min have elapsed since the failure of a content transfer process for the CDS object's content binary and resume content transfer process has not started.

M	C	DMS	M-DMS	n/a	[52] [55]

This guideline explains when a MediaServer needs to automatically destroy CDS objects that were created from upload AnyContainer or OCM: upload content.

The guideline explicitly covers the case where resume content transfer, as described in 7.4.1.7.20, is supported.

7.4.1.7.28.4 A UPnP AV MediaServer must not automatically destroy a CDS object or any of its `<res>` element when the MediaServer is doing a content transfer process for the CDS object.

M	A	DMS	M-DMS	n/a	[52] [55]

This guideline requires MediaServer implementations not to destroy CDS objects or `<res>` elements involved in an upload system usage.

As implied from the comment for 7.4.1.7.28.1, a MediaServer might choose to destroy such an object as a result of a CDS:DestroyObject request that was invoked from another control point. This type of a scenario is an unexpected user-initiated and is outside the scope of the ideal network conditions assumption.

7.4.1.7.28.5 If a UPnP AV MediaServer partially completes a content transfer process for a <res> element and res@dlna:resumeUpload="1", then the MediaServer must keep all res@importUri and res@dlna:importIfoFileURI that are associated with the CDS object valid for at least another 35 min.

M	A	DMS	M-DMS	n/a	[52] [55]

This guideline clarifies that MediaServers needs to preserve the validity of URI values that are used for uploaded IFO files and content binaries.

After a MediaServer partially receives a content binary, the device needs to extend the validity of the URI values (that have not completed a content transfer process) for at least 35 min. This provides a reasonable amount of time for an Upload Controller to retry or set up the next content transfer process.

See 7.4.1.7.20.4, 7.4.1.7.20.8 and 7.5.5.6.3 about resuming a failed content transfer process.

7.4.1.7.28.6 A UPnP AV MediaServer control point may use an OCM: destroy item.operation to cancel the following operations: upload AnyContainer or OCM: upload content.

O	C	+UP+	M-DMU	MU	[52] [55]

An upload controller uses this operation to signal the DMS to undo its previous operations that resulted in the creation of a CDS object.

7.4.1.7.29 MM/CM – Content validation and advertisement

7.4.1.7.29.1 If a UPnP AV MediaServer successfully completes a content transfer process for a CDS resource that was created with the DLNA.ORG_PN parameter in the 4th field, then the MediaServer may advertise that content using the supplied DLNA media format profile.

Advertise that content is meant specifically that the MediaServer provides a URI value for the <res> element. Before the MediaServer receives the content binary, the <res> element (without a URI value) will be advertised along with the res@importUri.

O	A	DMS	M-DMS	n/a	[52] [55]

DLNA UPnP AV MediaServers that support the upload device option can assume that control points that claim content conforms to DLNA are actually going to upload DLNA-conformant content. However, it is still good practice for a UPnP AV MediaServer to examine the uploaded content against the profile definition before advertising the content as being DLNA conformant.

7.4.1.7.29.2 If a UPnP AV MediaServer receives a content binary for a CDS resource that does not have the DLNA.ORG_PN parameter and it intends to advertise the content, then the MediaServer must validate the content before advertising it with the DLNA.ORG_PN parameter.

M	C	DMS	M-DMS	n/a	[52] [55]

Although an algorithm for validation is not specified, a UPnP AV MediaServer that determines that uploaded content is conformant to a DLNA media format profile is permitted to advertise it as such. The guideline does not obligate a DMS to either support uploading or advertising of non-DLNA content.

7.4.1.7.29.3 A UPnP AV MediaServer may change the value of a res@protocolInfo as part of the content validation process.

O	C	DMS	M-DMS	n/a	[52] [55]

7.4.1.7.29.4 If a UPnP AV MediaServer exposes uploaded content, then it should examine the uploaded content binaries to ensure that the metadata reported through the CDS is consistent with the content binary. In cases where a conflict is found, a UPnP AV MediaServer should appropriately correct the metadata reported through CDS.

S	A	DMS	M-DMS	n/a	[52] [55]

This is a good behavioural practice and helps to prevent interoperability problems.

7.4.1.7.30 MM/CM – General rules for CDS:DestroyObject

If a UPnP AV MediaServer implements CDS:DestroyObject, then a CDS:DestroyObject request on a CDS object, with the @dlna:dlnaManaged attribute, must result in a complete and atomic removal of the CDS item.

If the CDS:DestroyObject is a success, then the entire CDS object is removed from the CDS hierarchy.

If the CDS:DestroyObject is a failure, then the entire CDS object remains unaltered. Partial success (such as removal of some <res> elements or metadata properties) is unacceptable.

M	L	DMS	M-DMS	n/a	[52] [55]

Partially destroying a CDS object is problematic because control points have no way to recover the partially lost state.

The related question of whether the actual content binaries are deleted from the local storage is vendor-defined.

7.4.2 Image printing media management

7.4.2.1 General

Subclause 7.4.2 covers the guidelines for implementing image printing media management using the UPnP Printer architecture. Annex D also provides much useful information for implementers that want to implement the printing-related usages.

7.4.2.2 General capability requirements – MM UPnP printer compliance

DLNA Device Classes and Device Capabilities must fully support the mandatory portions of the UPnP Printer:1 specifications.

M	R	DMP+PR1+ +PR2+	n/a	n/a	[56] [57] [58] [59]

UPnP Printer:1 is the baseline architecture for discovering image printer devices.

7.4.2.3 Device requirements

7.4.2.3.1 MM UPnP printing controller-1 definition

The +PR1+ device capability must implement a UPnP Printer control point that uses the PE:CreateURIJob action to supply print jobs that allows selected image content to be printed.

M	L	+PR1+	n/a	n/a	[56] [57] [58] [59]

This guideline indicates that the +PR1+ device capability must use a UPnP control point that controls a UPnP printer to print images it has locally.

+PR1+ is recommended to delete the XHTML file after its use (refer to 7.4.2.3.5).

7.4.2.3.2 MM UPnP printing controller-2 definition

7.4.2.3.2.1 The +PR2+ device capability must implement a UPnP Printer control point that uses the PE:CreateURIJob action to supply print jobs that allows selected image content to be printed.

M	L	+PR2+	n/a	n/a	[56] [57] [58] [59]

This guideline indicates that the +PR2+ device capability must use a UPnP control point that controls a UPnP AV MediaServer to find image content and a UPnP Printer to print those images found on a UPnP AV MediaServer.

+PR2+ is recommended to delete the XHTML file after its use (refer to 7.4.2.3.5).

7.4.2.3.2.2 The +PR2+ device capability must implement a UPnP AV MediaServer control point for browsing a ContentDirectory service on a DMS or M-DMS to obtain image content to be printed.

M	R	+PR2+	n/a	n/a	[52] [55]

7.4.2.3.3 MM DMP+ UPnP printer device definition

A DMP+ must implement a UPnP Printer:1 device with the PrintEnhanced:1 service.

M	R	DMP+	n/a	n/a	[57] [58] [59]

DMP+ devices must implement the minimum baseline services for a UPnP Printer. Although the UPnP definition for a Printer:1 requires PrintBasic:1 and PrintEnhanced:1, the DLNA guidelines only define interoperability for the PrintEnhanced:1 service. A DMP+ will need to implement PrintBasic:1 if the implementation requires UPnP certification by the UPnP Implementers Corporation (UIC). In most cases, the implementation for a PrintBasic:1 service is the same underlying implementation used for a PrintEnhanced:1 service since PrintEnhanced:1 is a superset of PrintBasic:1.

7.4.2.3.4 MM DMP+ PrintEnhanced rules

A DMP+ must fully support the mandatory actions and state variables for a UPnP PrintEnhanced:1 service.

M	R	DMP	n/a	n/a	[59]

PrintEnhanced:1 adds several mandatory features over PrintBasic:1 that enables better photo printing capabilities such as the ability to print borderless photos.

7.4.2.3.5 MM XHTML-Print

7.4.2.3.5.1 A UPnP Printer device must specify the <dlina:X_DLNA CAP> element (as a child of the <device> element that represents the UPnP Printer device) that includes the printProfiles-AllRangeTrue-AllRangeFalse token

Table 21 – UPnP Printer dlna:X DLNACAP element

Capability ID	Description
printProfiles- AllRangeTrue - AllRangeFalse	Indicates the supported DLNA XHTML-Print document profiles for these scenarios: the XHTML-Print document only references images that support the HTTP Range header the XHTML-Print document references one or more images that do not support the HTTP Range header
More formally, the syntax of the capability ID is defined below. "printer-capability-id = "printProfiles" "-" all-range-true-profile "-" all-range-false-profile "all-range-true-profile = <DLNA XHTML profile name> "all-range-false-profile = <DLNA XHTML profile name> The "printProfiles" portion of the capability ID is a literal, case-sensitive string value.	
The all-range-true-profile is the top-most XHTML profile that the printer device supports when all images referenced by the XHTML-Print document support the HTTP Range header. The all-range-true-profile value must be XHTML_PT or a superset XHTML profile. The all-range-false-profile is the top-most XHTML profile that the printer device supports when one or more images referenced by the XHTML-Print document do not support the HTTP range header. The all-range-false-profile value must be equal to XHTML_Baseline or a superset XHTML profile. Given the hierarchy of XHTML profiles defined in [CROSSREF: MF printing class profile parameter sets profiles: All XHTML printing profiles, 10.1.1] the top-most XHTML profile is the profile that appears highest in the hierarchy, closest to the root XHTML profile of XHTML_ALL.	
A UPnP printer device that supports a DLNA-defined XHTML profile must adhere to these rules. • The printer device must be able to properly print any XHTML-Print document that conforms to the indicated XHTML profile definition. • The printer device must be able to properly print any XHTML-Print document that conforms to an XHTML profile that is a subset of the indicated XHTML profile. • Properly print is defined as the ability to print the XHTML-Print document in a manner consistent with all mandatory portions of [61] and [62] including but not limited to the document's instructions for image placement, image rotation, and borders. • The definition of properly print imposes these additional requirements on the content binary lengths (a.k.a. image file sizes). • If the XHTML profile is XHTML_Baseline and the referenced image does not support the HTTP range header, then the printer device must be able to print the image, only when its content binary length is equal to or less than 4194304 bytes (4 MB). • Otherwise, the printer device must be able to print each image, so long as the image meets all restrictions imposed by an image profile that is defined for use by the XHTML profile definition. This restriction applies regardless of whether the images support the HTTP range header or not. • The definition of support the HTTP Range header is defined by whether the op-param's b-val is "1", as obtained from – the image's 4th field value of res@protocolInfo value (acquired from CDS); – the contentFeatures.dlna.org value (acquired from an HTTP transport layer request that specified getContentFeatures.dlna.org)	

Example:

- <dlina:X_DLNACAP>
printProfiles-XHTML_Complex-XHTML_Baseline
</dlina:X_DLNACAP>

M	A	DMPPr	n/a	n/a	[61] [62] [75]

XHTML-Print defines a simple XHTML data stream suitable for printing. It is based on XHTML basic.

It should be noted that both PrintEnhanced:1 and PrintBasic:1 do not have any mechanism to automatically delete the XHTML-Print document (on the +PR1+ or +PR2+) when it is no longer needed. It is recommended that printer control points monitor the progress of submitted jobs and remember to delete the XHTML file after the job is completed, aborted, or never consumed. One way to do this is to monitor the evented variable PE.JobEndState and the presence of the printer.

The XHTML-Print/CSS print guidelines [75] should be referred to for information on how to compose consistent XHTML.

It should be noted that XHTML profile definitions define layout complexity and image profile definitions specify image complexity. The best way to avoid conflicts is for the XHTML profile to rely solely on the image profile definitions to define image limitations (for example, XHTML profiles should not impose restrictions for things like pixel resolution or content binary lengths, unless already defined by an existing printing-related specification such as [75]).

It should be noted that 7.5.5.2.24.4 requires a DMPPr to function correctly in scenarios where a DMS only supports a single HTTP connection.

7.4.2.3.5.2 If a UPnP Printer's PrintEnhanced SCPD document indicates support for PNG, then the printer must be able to properly print the PNG images that are referenced in an XHTML-Print document that conforms to an XHTML profile that the printer device supports.

Properly print and supports are as defined in 7.4.2.3.5.1 above.

Support for PNG is defined in the PrintEnhanced state variable PE.XHTMLImageSupported. The value of this element must be the DLNA-defined mime-type for PNG images (i.e., "image/png").

PNG images is defined as the set of permitted PNG image profiles defined in the XHTML profile definition.

M	A	DMPPr	n/a	n/a	[62]

7.4.2.3.5.3 A UPnP printer control point must not send XHTML-Print documents containing references to PNG images unless the Printer indicates support for PNG as defined in the guideline above.

M	A	+PR1+ +PR2+	n/a	n/a	[62]

7.4.2.3.5.4 UPnP printer control points must follow these restrictions when creating XHTML-Print documents that are used with a UPnP printer device:

- the XHTML-Print document used with a UPnP printer device must conform to a valid DLNA XHTML profile that is supported by the UPnP printer device, as indicated by the printProfiles-AllRangeTrue-AllRangeFalse token;
- if the XHTML profile is XHTML_Baseline and the referenced image does not support the HTTP range header, then the UPnP printer control point must ensure that the content binary length of the image is equal to, or less than, 4194304 bytes (4 MB);
- otherwise, the images referenced must comply with the requirements of the XHTML profile used. (In this case, the size restrictions on the referenced images are determined by the image profile definitions.)

Used with a UPnP printer device means that the printer control point instructed the printer device to print the XHTML-Print document and the images referenced in the document.

M	L	+PR1+ +PR2+	n/a	n/a	[62]

Since all XHTML profiles are organized hierarchically, an XHTML-Print document that conforms to a derived XHTML profile will automatically conform to the parent XHTML profile because derived profiles inherit all restrictions of parent profiles. For example, a document that conforms to XHTML_PT automatically conforms to XHTML_Complex.

Control points can acquire the file size in a variety of ways.

- by storing or serving the file directly;
- using res@size metadata;
- issuing an HTTP/1.0 HEAD request;
- by actually transferring the entire file to check its length, although this is not practical for large files.

7.4.2.3.5.5 UPnP printer control points must not use Multi-part MIME encoding for the XHTML-Print document.

M	L	+PR1+ +PR2+	n/a	n/a	[43] [62]

It is very difficult to guarantee that the printer will be able to print all of the included images, when encoded with multi-part MIME.

PrintEnhanced:1 requires the printer to support multi-part MIME but it is not recommended for applications in the future.

7.4.2.3.6 MM photo content selection

UPnP printer control points should allow the user to select the photos that they want to print and a choice of layouts.

S	A	+PR1+ +PR2+	n/a	n/a	[56] [57] [58] [59] [63]

XHTML-Print photo templates are available for developer's convenience. Refer to [63] for more information.

7.4.2.3.7 MM printer selection

UPnP printer control points must be capable of allowing the user to select the printing endpoint that they want to print to. This allows for the case when there is more than one image printer on the network.

M	A	+PR1+ +PR2+	n/a	n/a	[56] [57] [58] [59]

7.4.2.3.8 MM printer status monitoring

UPnP printer control points should monitor the status of the printer using the UPnP state variables and indicate to the user when there needs to be intervention (for example, paper out).

S	A	+PR1+ +PR2+	n/a	n/a	[56] [57] [58] [59]

7.4.2.3.9 MM acquiring image content URIs

7.4.2.3.9.1 UPnP printer control points must be able to select image content and insert their associated URIs into a XHTML-Print document from images it can access locally.

M	R	+PR1+	n/a	n/a	[56] [57] [58] [59] [62]

These guidelines indicates where image content URIs to be printed are located.

If a device class wants to support both the 2-box and the 3-box printing system usages, it must support both the +PR1+ and +PR2+ device capabilities.

7.4.2.3.9.2 UPnP printer control points must be able to select image content and insert their associated URIs into a XHTML-Print document from images it can access from a UPnP AV MediaServer.

M	R	+PR2+	n/a	n/a	[56] [57] [58] [59] [62]

7.5 Media transport

7.5.1 General

7.5 covers the requirements for the media transport layer. Table 22 summarizes the types of media transfer now available. The DLNA transfer mode terms are consistent with those found in 6.3 of [74].

Table 22 – DLNA Media transfer modes

Transfer mode	Transfer rate	Example usages	Default DLNAQOS level
Streaming transfer	The content source and content receiver maintain an average transfer rate equal to the rate required by the internal timing information of the content binary An example of the internal timing information of a content binary is the samples per second rate of an audio stream or the video bitrate of an AV content binary	Immediate rendering by the content receiver of content binaries with an inherent time base (for example, audio or AV media) Real-time generated AV media transfer followed by store/record at the content receiver	DLNAQOS_2
Interactive transfer	The content source transfers the content binary as fast as the content receiver can consume it without degrading any streaming transfer originating from the content source. The content source is not required to transmit as fast as the content receiver is able to consume it	Immediate rendering by the content receiver of content binaries with no inherent time base (for example, images or printer documents)	DLNAQOS_1
Background transfer	The content source transfers the content binary at a rate determined by the content source, but no faster than the rate at which the content receiver can accept the content binary from the network	Transfer and store of file-based media	DLNAQOS_0 (lowest priority)

Table 23 summarizes the combinations of permitted DLNAQOS priorities and transfer modes for each media class. The relationship between the different columns is described here.

- Media class: Indicates a media class.
- Transfer mode: Indicates a transfer mode.
- Combination permitted: Indicates if the indicated media class and transfer mode values can be combined. The "Yes" and "No" values indicate if the combination is permitted. A "default" value means that content sources are required to support the combination. The permissible combinations are described in the following guidelines.

7.4.1.3.34 MM-tm-s (Streaming Mode Transfer Flag)

7.4.1.3.35 MM-tm-i (Interactive Mode Transfer Flag)

7.4.1.3.36 MM-tm-b (Background Mode Transfer Flag)

- Permitted DLNAQOS_UP: Indicates the DLNAQOS_UP values that the content source is permitted to use when responding to transport requests. The guidelines do not require content sources to always respond with the highest DLNAQOS_UP value that is listed in this column. The following guidelines describe the permitted DLNAQOS_UP values for a given media class.

7.5.4.2.3.2 in 7.5.4.2.3 MT transfer mode

7.5.4.2.10 MT DLNAQOS background

7.5.4.2.11 MT DLNAQOS interactive transfer

7.5.4.2.12 MT DLNAQOS streaming

Table 23 – Permitted combinations of DLNAQOS_UP and transfer mode per media class

Media class	Transfer mode	Combination permitted	Permitted DLNAQOS_UP
Image, media collection file or XHTML print document	Streaming	No	n/a
	Interactive	Default	DLNAQOS_1, DLNAQOS_0
	Background	Yes	DLNAQOS_0
Audio	Streaming	Default	DLNAQOS_2, DLNAQOS_1, DLNAQOS_0
	Interactive	No	n/a
	Background	Yes	DLNAQOS_0
AV	Streaming	Default	DLNAQOS_2, DLNAQOS_1, DLNAQOS_0
	Interactive	No	n/a
	Background	Yes	DLNAQOS_0

A streaming transfer supports two media usages. The first case is where a content binary is being immediately rendered for a user and contains inherent timing that must be met. The second case is where a content binary is being generated in real time at a fixed rate (such as a live broadcast stream), regardless of whether the item is being immediately rendered or stored for later use. In either of these cases, a delay in packet delivery can adversely impact the user's perception of the system. If the content binary contains inherent timing information (such as audio or AV content) and is being immediately rendered, a delay in packet delivery may cause data to not be available on the content receiver at the time it needs to be played. This may lead to a dropout in the playback of the media. If a real-time stream is being generated on the content source and the throughput across the network (which can be affected by the content receiver's use of flow control) is not equal to the data rate of the generated content, a buffer overrun may occur on the content source. This may lead to a loss of data in the content binary.

NOTE If the content is being generated at an average fixed rate, such as the capture of audio or AV content from an external source, and the network has a period where the throughput is less than the rate of generation of the content, the content source will typically buffer the data for sending at a later time. However, if this period lasts long enough, the content source may exhaust its ability to buffer the captured content. At this point, some data samples will be lost. If the content receiver is rendering the stream immediately, the user will perceive the loss of the data as skipped content samples. If the content receiver is storing the stream for later use, such as in a download operation, the stored content will have missing samples. Content sources and receivers may have any amount of buffering to mitigate this situation and the network cannot be controlled to guarantee a bandwidth for the transfer. Any content that is transitory (i.e., not stored on disk) and may cause a buffer overrun on the content source due to network delays should always be sent as a streaming transfer.

An interactive transfer is used for the case where a content binary is being immediately rendered for a user but it does not have any inherent timing information, such as image media. In this case, a delay in packet delivery of a few milliseconds will not cause an adverse impact for rendering, but sufficiently long delays will adversely impact the user's perception of the system.

A background transfer is used for the case where the content binary is not being transferred for immediate rendering or where the user may be satisfied with a transfer executed at the lowest priority. It is typically reserved for the download or upload of content that is not being generated in real time by the content source. For example, this transfer mode would be used for downloading a content binary that has been stored in a file on the content source.

The content source is free to internally prioritize the background transfer lower than other transfer modes.

The DLNA QoS levels are shown in the above tables as an indication that each of the transfer modes is handled at a different QoS level. Further discussion of QoS can be found in 7.2.

Each of the transfer modes is implemented differently for the transport chosen (HTTP versus RTP). In addition, a transport may choose to not allow a particular transfer mode. For instance, RTP will not support background transfers. The transfer modes available to a particular content resource are specified in 7.4.

For most transfers, the content receiver issues a request for the content binary with a specific transfer mode and media transport based on the type of usage involved. An upload content transfer is the exception to this rule. It is initiated by the content source.

However, the choice of transfer mode is made in the same way, based on the type of usage and the content binary to be transferred.

The definitions of the transfer modes introduced in this clause apply only to a state of the network referred to as ideal network conditions. Under ideal network conditions no congestion from other communicating parties or from bandwidth restrictions on the network exists. Furthermore, under ideal network conditions, content sources and content receivers have moved away from the start-up conditions, and have reached steady-state transfers. From the perspective of measurements, ideal network conditions can be approximated by having a single content source send data to a single content receiver over a high-speed link with bandwidth equal to, or exceeding, the bandwidth required by the transmission.

This system provides a robust set of transfer modes that support streaming for immediate rendering or download of content for later consumption. It also provides a structure for fine grained QoS control.

7.5.2 Uniform client data availability model

The uniform client data availability model (UCDAM) provides a mechanism for describing the data available for a content stream. It defines a content stream in mathematical terms, with special attention focused on the data range that can be transmitted by the content source. The model applies regardless of whether the content is stored, converted (for example, transcoded, transrated, transscaled, etc.) or "live". Although the DLNA guidelines do not specify buffering implementations or requirements for either the content source or the content receiver, the uniform data availability model takes into account the diversity of buffering models that are available to implementers. These guidelines can then focus on normative high level behaviours that are common, regardless of details at the transport layer. Figure 18 graphically shows the UCDAM model.

[d_x, d_y] : Informative

- Represents the entire stream
- Beginning may be undefined
- May be infinitely long

[s_0, s_N] : Normative

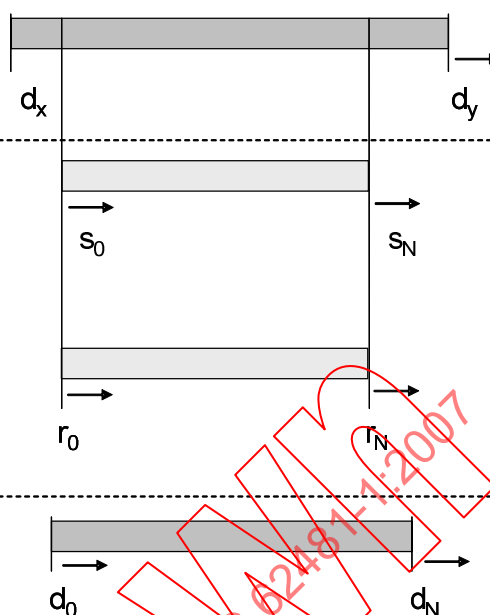
- Content source is able to transmit this data range on the network
- Data range can slide and/or grow

[r_0, r_N] : Normative

- Content source can perform random access requests on this data range
- If random access is supported, then this range is always equal to [s_0, s_N]

[d_0, d_N] : Informative

- Content Receiver has access to this data range
- Data range can slide and/or grow
- Content receiver can store/cache/buffer data such that [d_0, d_N] != [s_0, s_N]



IEC 1586/07

Figure 18 – UCDAM summary

- The first aspect of the UCDAM is the definition of a content stream. A content stream is media with a beginning (d_x) and end (d_y), where both values are defined by the content source. In some cases, a content stream never ends (d_y increases with time). This data range is an assumed condition but is not referenced in the guidelines.
- The second aspect of the UCDAM is the data range available from the content source for transport to the content receiver. This range is denoted as [s_0, s_N]. For content stored within a file on the content source, s_0 may be equal to d_x and s_N may be equal to d_y . For content that may be captured by the content source from a live AV or audio feed, the range represents the amount of data buffered by the content source. This data range is normative and may be referenced in the guidelines.
- The third aspect of the UCDAM is the data range available to the content receiver. The range of data available to the content receiver, defined inclusively as [d_0, d_N], is determined by two aspects: what the content source can transmit (i.e. some data range of [s_0, s_N]) and, in addition, what may be buffered on the content receiver in a local manner². This data range is an assumed condition but is not referenced in the guidelines.
- The fourth aspect of the UCDAM is the content source's data range that supports random access requests. The [r_0, r_N] data range represents the data range where random access operations are supported. If this capability is supported, then the [r_0, r_N] interval is always equal to the [s_0, s_N] data range. If the content source does not allow random access within the content stream, then a content receiver can only request content starting from s_0 . This data range is normative and may be referenced in the guidelines.
- The fifth aspect of the UCDAM is that the data range available to the content receiver can change with time. This is really a clarification of the third aspect. As time passes, the data range that the content source can provide to the content receiver can also change. There are three aspects that determine how the data range available to the content receiver will change.
 - Does the content source guarantee that s_0 is fixed or can s_0 increase with time? If s_0 is fixed, then the content source is characterized as operating under a fixed- s_0 model.

² Neither the UCDAM model, nor the DLNA media transport guidelines specify how clients have access to data in a local manner as implementers may use a variety of memory-based and disk-based mechanisms to define the range of data that a content receiver can access without the content source having to transfer data.

Otherwise, the content source is characterized as operating under an increasing- s_0 model. An example of a possible fixed- s_0 model is content data read from a file on the content source. An increasing- s_0 model may be used to represent content being captured from an incoming AV feed.

- Does the content source guarantee that s_N is fixed or can s_N increase with time? If s_N is fixed, then the content source is characterized as operating under a fixed- s_N model. Otherwise, the content source is characterized as operating under an increasing- s_N model. The same examples as above can be used in this case. An example of a fixed- s_N model is content data read from a file on the content source. An increasing- s_N model may be used to represent content being captured from an incoming AV feed.³
- Does the content receiver save data ⁴ such that ($d_0 < s_0$)? (i.e., the content receiver has accessible data that the content source can no longer provide).

This model is very flexible in that it can easily represent all of the common types of media streams. For example, a media stream originating from a file can be represented as $d_0 = dx$ and $dN = dY$ – the entire extent of the content is available to the user. An unbuffered live stream can be represented as $d_0 = d_N$, so that only the current moment in time is available. In addition to above simple examples, many more complex buffering systems can be represented by UCDAM. See Annex E for more details.

Most of the UCDAM model is conceptual and outside the scope of the DLNA guidelines. However, the $[s_0, s_N]$ and $[r_0, r_N]$ data ranges describe what data can be requested in media transfer operations over the network, and as such are within the scope of these guidelines. Therefore, those data ranges will be utilized in the following guidelines where appropriate to define the range of content that can be transferred using DLNA-defined media operations.

7.5.3 Media operations

A media operation is the network level operation that supports a user interaction with a content binary. At a high level, they define the network operations that must occur in order to support a streaming transfer mode data transfer. The core set of media operations that an endpoint can perform are defined in the following table.

Table 24 – DLNA streaming media operation definitions

Term	Definition
Play	Initiate a streaming transfer for playback of media. The transfer occurs at a rate that supports normal playback of the content binary. The transfer begins at s_0 and proceeds at a rate sufficient to support normal (1x) playback of the content binary. The operation completes after transfer of a fixed s_N value. Under the increasing- s_N model the transfer does not complete
Stop	Terminate a streaming transfer
Pause	Temporarily suspend a streaming transfer
Pause-Release	A pause media operation has suspended a streaming transfer, completed the pause media operation and re-established the flow of data over the network to support the streaming transfer

³ The examples here are not intended to imply that the content source must use a particular model when representing file-based or captured media, but should be regarded as illustrative of one possible implementation.

⁴ Neither the UCDAM model, nor the DLNA media transport guidelines require or specify how receiving endpoints save data.

Table 22 (continued)

Term	Definition
Seek	Move the transfer point to a particular point in a stream in the range $[r_0, r_N]$, that represents the seek-able range. (The seek-able data range is the same random access data range, although the former term is used for streaming transfers and the latter term applies to streaming, interactive, and background transfers.) If a seek-able range exists, it must equal $[s_0, s_N]$. The next set of transferred data will be from the indicated point in the content binary. DLNA defines two types of seek operations: – byte-based seek: a seek operation where the transfer point is specified in units of bytes; – time-based seek: a seek operation where the transfer point is specified in units of time
Fast forward scan	Perform data transfers that will support a positive play speed greater than 1x
Slow forward scan	Perform data transfers that will support a positive play speed greater than 0 but less than 1x
Fast backward scan	Perform data transfers that will support a negative play speed less than -1x
Slow backward scan	Perform data transfers that will support a negative play speed less than 0 but greater than -1x
Streaming download	Initiate a streaming transfer for the purpose of storing media for later playback. The transfer begins at s_0 and proceeds at a rate defined by the internal timing information of the content. The operation completes after transfer of a fixed s_N value. Under the increasing- s_N model the transfer does not complete until the content receiver terminates the transaction

The listed media operations are defined in terms of how the media transfer occurs over the network for a given transport protocol in the guidelines below.

7.5.4 Media transport protocols

7.5.4.1 General

The following subclauses contain guidelines for the use of media transports in DLNA devices. The guidelines are organized into a subclause that covers requirements common among all media transports and then subclauses that cover requirements for specific media transport protocols such as HTTP and others.

7.5.4.2 Media transport common guidelines

7.5.4.2.1 MT mandatory transport support

Content sources and content receivers must support HTTP as the mandatory transport as specified in the following subclauses.

- HTTP media transport common guidelines (7.5.5.2.1 to 7.5.5.2.38)
- HTTP media transport for streaming transfer (7.5.5.3.1 to 7.5.5.3.19)
- HTTP transport – Interactive transfer (7.5.5.4.1 to 7.5.5.4.3)
- HTTP transport – Background transfer (7.5.5.5.1 to 7.5.5.5.3)
- HTTP transport – POST (7.5.5.6.1 to 7.5.5.6.5)

M	A	DMP DMR DMP+ DMS +PU+ +UP+ +DN+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	n/a

7.5.4.2.2 MT optional transport support

Content sources and content receivers may support RTP as an optional transport as specified in the following subclauses: 7.5.6.2.1 through 7.5.8.2.80.

O	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	n/a

7.5.4.2.3 MT transfer mode support

7.5.4.2.3.1 A media transport protocol may limit the transfer modes that are supported for a content binary.

O	A	DMP DMR DMPr DMS +PU+ +UP+ +DN+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	n/a

For example, RTP may be only used for streaming transfer modes, while HTTP can support all three available modes.

7.5.4.2.3.2 An endpoint initiating a media transfer request must specify a transfer mode appropriate for a given content binary, and transport protocol used.

The list of available transfer modes for a media class is as specified below. This may be further limited by the transport protocol chosen; see 7.5.4.2.3.1.

Table 25 – MT media class transfer modes

Media class	Transfer mode
XHTML print doc:	Background, interactive
Media collection binary:	Background, interactive
Image:	Background, interactive
Audio-only:	Background, streaming
AV:	Background, streaming

M	A	DMP DMR DMPr +UP+ +DN+	M-DMP M-DMD M- DMU	MIU	n/a

This can be either a content receiver requesting a particular transfer mode for content that it will receive or it can be a content source specifying the transfer mode when performing an upload operation.

7.5.4.2.3.3 A content source may constrain the transfer of audio-only or AV content binaries to streaming transfer only if it is not able to support the background transfer mode for that content binary and transport protocol.

O	A	DMS +PU+	M-DMS	n/a	n/a

For example, a content source may do this because it is capturing content from a live stream and could potentially overflow its buffer if the network transfer is handled below the nominal rate of the content.

7.5.4.2.3.4 A content source must indicate the media transfer modes that are available for a content binary by setting the tm-s, tm-i, and tm-b flags in the 4th field of the protocolInfo.

M	A	DMS +PU+	M-DMS	n/a	n/a

See 7.4.1.3.23.2, 7.4.1.3.34.1, 7.4.1.3.35.1 and 7.4.1.3.36 for more information.

7.5.4.2.3.5 An endpoint initiating a media transfer should investigate if the desired transfer mode is available before the initiation of the transfer via the 4th field of the protocolInfo.

S	A	DMP DMR DMP+ +UP+ +DN+	M-DMP M-DMD M- DMU	MIU	n/a

See 7.4.1.3.23.2, 7.4.1.3.34.1, 7.4.1.3.35.1 and 7.4.1.3.36 for more information.

7.5.4.2.3.6 An endpoint responding to the initiation of a media transfer must generate an appropriate transport protocol error response if the requested transfer mode is not currently available for the given content binary.

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	n/a

This can be either the content source responding to an incorrectly requested transfer mode by a content receiver or in the case of an upload operation it can be the content receiver responding to an incorrectly requested transfer mode by a content source. See 7.5.5.4.1.4, 7.5.5.4.2.2, 7.5.5.5.1.5, and 7.5.5.5.2.2 for HTTP error codes used in various scenarios.

Subclause 7.5.4.2.3.2 defines the possible transfer mode given the media type and 7.5.4.2.3.3 allows content sources to constrain the available transfer mode for a given item (for example, if RTP is used as the transport protocol). This guideline requires an endpoint to generate an error if the request for the content is not of the allowed modes for the media item.

7.5.4.2.4 MT transfer mode support for device classes

7.5.4.2.4.1 A DMS, M-DMS, or +PU+ that supports the AV or audio media classes must support acting as a content source for streaming transfers.

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.4.2.4.2 A DMS, M-DMS, or +PU+ that supports the image media class must support acting as a content source for interactive mode transfers.

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.4.2.4.3 A DMS or M-DMS that supports the download system usage must support acting as a content source for background mode transfers.

M	A	DMS	M-DMS	n/a	n/a

The requirement is not that background must be used for the transfer. Rather, the guideline states that the background transfer mode must be supported for one or more content binaries. See 7.4.1.3.36 MM tm-b for more information about reporting background transfer mode support for a content binary.

7.5.4.2.4.4 A DMS or M-DMS that supports the upload AnyContainer or OCM: upload content operations must support acting as a content receiver for background mode transfers as part of the content transfer process.

M	A	DMS	M-DMS	n/a	n/a

7.5.4.2.4.5 A DMS or M-DMS that supports the upload AnyContainer or OCM: upload content operations should support acting as a content receiver for streaming and interactive mode transfers as part of the content transfer process.

S	A	DMS	M-DMS	n/a	n/a

Allows the server to support the upload of content that cannot be sent via background transfer mode such as live content captured from a tuner.

7.5.4.2.4.6 A DMP, M-DMP, or DMR that supports the audio or AV media classes must support acting as a content receiver for streaming mode transfers.

M	A	DMP DMR	M-DMP	n/a	n/a

7.5.4.2.4.7 A DMP, M-DMP, or DMR that supports the image media class must support acting as a content receiver for interactive mode transfers.

M	A	DMP DMR	M-DMP	n/a	n/a

7.5.4.2.4.8 A DMP_r must support acting as a content receiver for interactive mode transfers.

M	A	DMP _r	n/a	n/a	n/a

7.5.4.2.4.9 An M-DMD or +DN+ must support acting as a content receiver for background mode transfers.

M	A	+DN+	M-DMD	n/a	n/a

The requestor is not obligated to use background transfer if the server defines that only streaming of this content is supported.

7.5.4.2.4.10 An M-DMD or the +DN+ may support acting as a content receiver for streaming mode transfers.

O	A	+DN+	M-DMD	n/a	n/a

Supports the download of content that cannot be sent via background transfer mode.

7.5.4.2.4.11 An M-DMU or +UP+ must support acting as a content source for background mode transfers.

M	A	+UP+	M-DMU	n/a	n/a

7.5.4.2.4.12 A +PR1+ or +PR2+ must support acting as a content source for interactive mode transfers.

M	A	+PR1+ +PR2+	n/a	n/a	n/a

XHTML documents and images are transferred under interactive mode by default.

7.5.4.2.5 MT low throughput tolerance

Content receivers must tolerate scenarios where an HTTP server endpoint is not able to sustain a particular transmission throughput.

Tolerate means that the content receiver is able to do one of the following things gracefully (i.e., without crashing or requiring the user to power-cycle or reset the device):

- continue receiving content data; or
- terminate the transport layer connection.

M	A	DMP DMR DMPr DMS +DN+	M-DMS M-DMP M- DMD M-DMU	MIU	n/a

This guideline is mandatory because a home network does not always operate under ideal network conditions (i.e., the transmission rate remains dependent on the network throughput between the server and client). Products that crash, require a reset, or a similar type of power-cycle operation as a result of low transmission throughput violate this guideline.

DLNA devices are permitted to have user interactions in meeting the tolerance portion of this requirement. For example, a DMP is permitted to report to ask the user if they want to stop rendering or a download because the throughput is extremely slow.

7.5.4.2.6 MT requirements for background transfer

If background transfer is available for a given content binary, a downloading endpoint (an M-DMD, or +DN+) must use the background transfer mode when performing a download operation.

M	A	+DN+	M-DMD	n/a	n/a

Examples of where this wouldn't be the case are the downloading of live media streams. On download, the server will mark them as only transportable with streaming transfer mode

7.5.4.2.7 MT streaming transfer rate assumptions

7.5.4.2.7.1 A content receiver operating in streaming transfer mode must be able to receive content from the network at rates required to sustain streaming transfer for the profiles that the content receiver supports.

M	A	DMP DMR +DN+	M-DMP M-DMD	MIU	n/a

A content receiver must be able to receive and consume content at a rate that will allow it to render the content in real-time. This guideline does not apply where the content source and content receiver have negotiated transfer characteristics within the transport protocol, such as negotiated buffer agreements within RTP. This guideline applies only in the absence of an existing agreement between the content receiver and the content source.

7.5.4.2.7.2 A content source operating in streaming transfer mode must be able to send content to the network at rates required to sustain streaming transfer for the content binary.

M	A	DMS +PU+ +UP+	M-DMS M-DMU	n/a	n/a

7.5.4.2.8 MT interactive transfer rate assumptions

7.5.4.2.8.1 Content sources and content receivers using an interactive transfer must tolerate all transmission throughputs.

In this case, tolerate means that the content source is not permitted to terminate the transport connection simply because the throughput is too low, unless user intervention has caused it to happen.

M	A	DMS DMPr DMP DMR +PU+ +DN+ +PR1+ +PR2+	M-DMS M-DMD	MIU	n/a

For example, sending an image over the network for rendering can occur at any rate depending on the network and server load.

7.5.4.2.8.2 Content sources and content receivers using an interactive transfer may affect the actual rate of data delivery using transport layer flow control, regardless of the content's internal timing information.

O	C	DMS DMP +PU+ +PR1+	DMR +DN+ +PR2+	M-DMS	M-DMD	MIU	n/a

7.5.4.2.8.3 For interactive transfers, a content receiver must not make any requirements on the actual rate at which the content source makes data available to be sent over the network.

M	A	DMP +DN+	DMR	M-DMP	M-DMD	MIU	n/a

7.5.4.2.9 MT background transfer rate assumptions

7.5.4.2.9.1 Content sources and content receivers using a background transfer must tolerate all transmission throughputs.

In this case, tolerate means that the content source is not permitted to terminate the transport connection simply because the throughput is too low, unless user intervention has caused it to happen.

M	A	DMS +DN+	DMP +UP+	M-DMS DMU	M-DMD	M-	MIU	n/a

For example, downloading of audio-only or AV content sourced from a file on the content source can occur at any rate (for example, higher or lower than the internal timing information of the content data).

7.5.4.2.9.2 Content sources and content receivers using a background transfer may affect the actual rate of data delivery using transport layer flow control, regardless of the content's internal timing information.

O	C	DMS +DN+	DMP +UP+	M-DMS DMU	M-DMD	M-	MIU	n/a

7.5.4.2.10 MT DLNAQOS background transfer

7.5.4.2.10.1 If DLNAQOS as defined in 7.2 is implemented, background transfer requests must be tagged with DLNAQOS_0 in accordance with Table 8.

M	R	DMS	DMP +DN+	M-DMS	M-DMD	MIU	n/a

7.5.4.2.10.2 If DLNAQOS as defined in 7.2 is implemented, Background transfers of content binaries must be tagged with DLNAQOS_0 in accordance with Table 8.

M	R	DMS	+PU+ +UP+	M-DMS	M-DMU	n/a	n/a

7.5.4.2.10.3 If DLNAQOS as defined in 7.2 is implemented by a content source and it receives a background transfer request for content that it cannot transfer at DLNAQOS_0, then it must respond with an error within the transport used, at DLNAQOS_0, in accordance with Table 8.

For HTTP as a transport, see 7.5.5.5.1.5 for the specific error.

M	R	DMS +PR1+ +PR2+	M-DMS	n/a	n/a

For example, a content receiver that tries to use a background transfer to acquire a live stream may receive an error response from the content source because it cannot transmit a live stream at DLNAQOS_0.

Note That +PR1+ and +PR2+ implement this guideline when they can only transmit XHTML documents and image files using the interactive transfer mode.

7.5.4.2.11 MT DLNAQOS interactive transfer

7.5.4.2.11.1 If DLNAQOS as defined in 7.2 is implemented, interactive transfer requests must be tagged with DLNAQOS_1, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), in accordance with Table 8, independent of the transport used. It should be noted that this guideline applies only when the transfer request is not marked as a background transfer.

M	R	DMPr	DMP	DMR	M-DMP	MIU	n/a

This transfer is part of an interactive experience and therefore the default is a higher QoS level than a background transfer.

7.5.4.2.11.2 If DLNAQOS as defined in 7.2 is implemented, interactive transfers of content binaries must be tagged with DLNAQOS_1, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), in accordance with Table 8, independent of the transport used.

M	R	DMS +PU+ +PR1+	M-DMS	n/a	n/a

7.5.4.2.12 MT DLNAQOS streaming media

7.5.4.2.12.1 If DLNAQOS as defined in 7.2 is implemented, streaming transfer requests must use DLNAQOS_2, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), in accordance with Table 8.

M	R	DMP	DMR	+DN+	M-DMP	M-DMD	MIU	n/a

For example, a client endpoint issues an HTTP GET request for AV content with DLNAQOS_2. The content source will then respond to the request with media that is tagged with DLNAQOS_2. The content source response (transfer of the actual media) will be tagged with DLNAQOS_2. Subsequent TCP ACK messages will use the existing TCP connection and therefore be tagged with DLNAQOS_2.

For the RTP transport, this priority is applicable to both audio and video streams encompassing TS, PS, and ES formats.

See 7.2.4.2.2 NC Ethernet DLNAQOS.

If DLNAQOS as defined in 7.2 is implemented, streaming transfer of content binaries must use DLNAQOS_2, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), in accordance with Table 8.

M	R	DMS +PU+	M-DMS	n/a	n/a

7.5.4.2.13 MT normative syntax for npt-time

The syntax of the npt-time token must be as follows:

- npt time = npt sec | npt hhmmss;
- npt sec = 1*DIGIT ["." 1*3DIGIT];
- npt hhmmss = npt hh ":" npt mm ":" npt ss ["." 1*3DIGIT];
- npt hh = 1*DIGIT ; any positive number;
- npt mm = 1*2DIGIT ; 0-59;
- npt ss = 1*2DIGIT ; 0-59.

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33]

This guideline provides a consistent syntax for NPT time positions for both DLNA's extensions to HTTP and RTP media transport.

7.5.4.2.14 MT normative random access data availability models

7.5.4.2.14.1 If a content source supports random access requests on a content binary, then the content source must use only one of the following random access data availability models.

- "Full random access data availability" model, as defined in 7.5.4.2.15.
- "Limited random access data availability" model, as defined in 7.5.4.2.16.

These random access data availability models must be used in a mutually exclusive manner.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	n/a

Other guidelines explain how to detect which random access data availability model is being used. Specifically, the op-param is tied solely to the "full random access data availability" model and the lop-npt/lop-bytes flags are tied solely to the "limited random access data availability". For more information, see the following guidelines.

7.4.1.3.18 MM op-oparm (operations parameter – common guidelines)

7.4.1.3.28 MM lop-npt and lop-bytes (limited operations flags): common

It should be noted that the "full random access data availability" model is the only model that can be used for content sources when serving image content or XHTML print documents. This limitation is inherent to the nature of such content, which neither have changing $[s_0, s_N]$ data boundaries nor have any sender-pacing requirements.

7.5.4.2.14.2 The UCDAM data range of $[s_0, s_N]$ must represent the entire data range that the content source can serve to other network endpoints.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	n/a

Although not testable by itself, these guidelines repeat normative portions of Figure 18. Other DLNA guidelines can normatively refer to the $[s_0, s_N]$ data range.

With regards to how the s_0 and s_N boundaries change, other DLNA guidelines explain how to represent these abstract data boundaries in terms of zero-based byte indices or npt playback positions.

Generally, the s_0 and s_N data boundaries increase with time, although in some scenarios the values can be reset, as is sometimes necessary when the integer value rolls over. Other guidelines specify the details on how these data boundaries can change.

7.5.4.2.14.3 The $[s_0, s_N]$ may change with time. Specifically, the following may happen.

- The s_0 data boundary may change with time.
- The s_N data boundary may change with time.

How these data boundaries change with time is undefined by the guidelines because these data boundaries are abstract. In some cases, other DLNA guidelines will impose restrictions that require a data boundary to remain fixed.

O	A	DMS +PU+ +PR1+	M-DMS	MIU	n/a

7.5.4.2.14.4 If a content source supports random access requests on content data, then the following rules must apply.

- The UCDAM data range of $[r_0, r_N]$ must represent the data range where random access operations are permitted.
- The $[r_0, r_N]$ data range must be equal to the $[s_0, s_N]$ data range.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	n/a

If random access requests are supported, then they need to be supported for the entire range that the content source can access.

This guideline is consistent with the following introductory material:

- Figure 18;
- the prerequisites used for the seek media operation definition in Table 22.

7.5.4.2.15 MT "full random access data availability" model

7.5.4.2.15.1 If a content source uses the "full random access data availability" model, then the following rules must apply.

- The entire content binary must be defined as the $[s_0, s_N]$ data range.
- The s_0 data boundary must map to a fixed and non-changing beginning. This requirement is a restriction of 7.5.4.2.14.3.
- The data range of $[s_0, s_N]$ must occupy an npt range of $[0, \text{npt-last-time}]$ and a byte range of $[0, \text{last-byte-pos}]$, where npt-last-time is in units of npt and last-byte-pos is in units of bytes.
- The content binary's zero position (i.e. npt-time=0 and byte-pos=0) must map to the UCDAM's data position of s_0 .

- The last-byte-pos and npt-last-time must map to the UCDAM's s_N data position and the s_N data boundary must map to the end of the available content data. (This requirement works in conjunction with 7.5.4.2.14.3.)
- The $[r_0, r_N]$ and $[s_0, s_N]$ data ranges must have the same equality.
- Responses to random access requests on the $[r_0, r_N]$ data range must be timely under ideal network conditions.

Timely means that the content source (under ideal network conditions) is able to begin responding with the requested content data within 27 s of receiving the request.

It should be noted that the npt-last-time, last-byte-pos, and byte-pos tokens for this guideline are relative to the complete content binary that is currently available rather than being relative to the content data returned in response.

M	A	DMS +PU+ +PR1+	M-DMS	MIU	n/a

This guideline defines the behavioural model for the "full random access data availability" model.

For guidelines on the data range for HTTP under "full random access data availability", see the following guidelines.

7.5.5.2.18 MT HTTP common random access data availability

7.5.5.2.19 MT HTTP – Data range of "full random access data availability"

The "relative to the complete content binary" phrase means that the tokens apply in the context of the whole content binary. It is incorrect to interpret these tokens as they are used in actual response data. For example, if last-byte-pos=100 then it is correct to conclude that the complete content binary currently has 101 bytes. It is not necessarily true that a response with the content-range header's last-byte-pos=50 means that the complete content binary has 51 bytes because the last-byte-pos token in this context simply means that the last byte in the entity body is the 51st byte of the complete content binary.

7.5.4.2.15.2 The values for npt-last-time and last-byte-pos (as specifically used in 7.5.4.2.15.1) may increase with time.

O	A	DMS +PU+	M-DMS	MIU	n/a

The concept of entire content range is relative to the current moment in time, in the context for the op-param. This means the s_N position can increase with time, causing the duration of the content binary to also increase (although the beginning has to remain fixed).

This model can apply when streaming content that is being recorded to local storage. The absolute beginning (s_0) never changes and as time passes, the end (s_N) increases.

7.5.4.2.16 MT "limited random access data availability" model

7.5.4.2.16.1 If a content source uses the "limited random access data availability" model, then only one of the two modes of operation must be used.

- Mode=0
- Mode=1

Furthermore, the following rules must be true for both Mode=0 and Mode=1:

- the $[r_0, r_N]$ and $[s_0, s_N]$ data ranges must have the same equality;

- the $[r_0, r_N]$ data range must be the limited data range.

M	A	DMS +PU+	M-DMS	MIU	n/a

This guideline explains the behaviour of the s_0 data boundary when it is used with the "limited random access data availability" model.

For guidelines on the data range for HTTP under "limited random access data availability", see the following guidelines.

7.5.5.2.18 MT HTTP common random access data availability

7.5.5.2.19 MT HTTP – Data range of "full random access data availability"

It should be noted that the limited data range is the data range that supports seek media operations, as clarified in the other rows of this guideline.

7.5.4.2.16.2 If a content source uses the "limited random access data availability" model under Mode=0, then the following must be true.

- The s_0 data boundary must map to a beginning that must change with time.
- The data range of $[s_0, s_N]$ must map to the npt range of $[npt\text{-}start\text{-}time, npt\text{-}last\text{-}time]$ and the byte range of $[first\text{-}byte\text{-}pos, last\text{-}byte\text{-}pos]$, where npt-start-time and npt-last-time are in units of npt and first-byte-pos and last-byte-pos are in units of bytes.
- There exists a "live position" that must be equal to the s_N data boundary.
- If the s_N data boundary is changing with time, then the "live position" must shift forward in real-time.
- Responses to random access requests on the $[r_0, r_N]$ data range must be timely under ideal network conditions.
- If the content source receives a transport layer request that is not a random access request (for example, HTTP request that omits range and TimeSeekRange.dlna.org) then the content source must respond with content data from the "live position". (See 0 for an example of how this guideline applies specifically to HTTP.)

Timely means that the content source (under ideal network conditions) is able to begin responding with the requested content data within 27 s of receiving the request.

Real-time is the data rate necessary for immediate rendering.

Note that the npt-start-time, npt-last-time, first-byte-pos, last-byte-pos, and byte-pos tokens for this guideline are relative to the complete content binary that is currently available, rather than being relative to the content data returned in response.

M	A	DMS +PU+	M-DMS	MIU	n/a

This guideline defines Mode=0 behaviours for the "limited random access data availability" model. This mode of operation is generally useful for live content streams that use a fixed data buffer that map to the $[s_0, s_N]$ and $[r_0, r_N]$ data ranges. Live television broadcast streams are ideal candidates for this data availability model.

The reason why $[r_0, r_N]$ is a limited data range in this context is that the values for npt-start-time, npt-last-time, first-byte-pos, and last-byte-pos change over time. For example, the value of first-byte-pos changes

- at time-0, the first-byte-pos is 1024. Random access requests that attempt to access before byte position 1024 will not work;

- at time-60, the first-byte-pos becomes 14749767106. Random access requests that attempt to access before byte position 14749767106 will not work, even though byte position 1024 was valid at time-60.

See the comment in 7.5.4.2.15 for help with interpreting the “relative to the complete content binary” phrase.

7.5.4.2.16.3 If using “limited random access data availability” Mode=0, then a content source must use increasing values for npt-start-time and first-byte-pos when reporting the available random access data range, unless one of the following conditions is true:

- 180 min have elapsed since the last transport layer request for the content binary;
- the value of last-byte-pos or npt-last-time causes a rollover because the maximum permitted value defined for the data type has been exceeded.

This guideline only applies when “limited random access data availability” applies to the scenario.

M	A	DMS +PU+	M-DMS	MIU	n/a

As time passes, the data range that is accessible can also change with time because the server's s_0 position can change.

Furthermore the s_N position can increase with time, usually causing the duration of the content binary to either temporarily increase when s_0 is temporarily non-changing or remain relatively constant over time when s_0 slides with s_N .

Given the nature of infinite streams (for example, live content), content sources are permitted to use a lesser values for first-byte-pos and npt-start-time.

The following example is an example timeline that exhibits this behaviour:

- at time-0, the first-byte-pos is 0;
- content requests occur for 180 min;
- at time-180, the first-byte-pos is 1474976710655;
- no content requests for 180 min;
- at time-360, the first-byte-pos is 0;
- content requests occur for 360 min;
- at time-720, the last-byte-pos exceeds 281474976710655, so the server changes first-byte-pos to 0.

It should be noted that a content receiver endpoint cannot seek to a position which it remembers if it does not request the content for over 180 min.

7.5.4.2.16.4 If a content source uses the “limited random access data availability” model under Mode=1, then the following must be true.

- The s_0 data boundary must map to a fixed and non-changing beginning. This requirement is a restriction of 7.5.4.2.14.3.
- The s_0 data position must be the static and absolute beginning for the content. (i.e., the s_0 position is the beginning of the content that does not change with time).
- The data range of $[s_0, s_N]$ must map to the npt range of $[0, \text{npt-last-time}]$ and the byte range of $[0, \text{last-byte-pos}]$, where npt-last time is in units of npt and last-byte-pos is in units of bytes.
- The content binary's zero position (i.e. npt-time=0 and byte-pos=0) must map to the UCDAM's data position of s_0 .

- The last-byte-pos and npt-last-time must map to the UCDAM's s_N data position and the s_N data boundary must map to the end of the available content data. (This requirement works in conjunction with 7.5.4.2.14.3.)
- Random access operations on $[r_0, r_N]$, where units are in npt, must be timely for the entire range of $[r_0, r_N]$.
- Random access operations on $[r_0, r_N]$, where units are in bytes, must be timely only for a limited subset of $[r_0, r_N]$. Random access operations outside this subset are guaranteed to be satisfied but timeliness is not guaranteed.
- If the content source receives a transport layer request that is not a random access request (for example, HTTP request that omits range and TimeSeekRange.dlna.org) then the content source must respond with content data from the beginning (i.e., npt=0 or byte-pos=0).

Timely means that the content source (under ideal network conditions) is able to begin responding with the requested content data within 27 s of receiving the request.

Note that the npt-last-time, npt-time, last-byte-pos, and byte-pos tokens for this guideline are relative to the complete content binary that is currently available, rather than being relative to the content data returned in response.

M	A	DMS +PU+	M-DMS	MIU	n/a

This guideline defines Mode=1 behaviours for the "limited random access data availability" model. Mode=1 is most useful for converted content, where content sources have access to a fixed beginning for the content but can lack the computational power to respond in a timely manner to byte-based random access positions. In such cases, the content source's indicated values for bytes-range represent the range where the server can provide a fast response, which is often the portion of converted content that is available at the current moment. Content receivers are permitted to request data outside those ranges (assuming the requested data falls within the $[s_0, s_N]$ data range, but the server may have a significant delay before responding.

It should be noted that live broadcast streams are generally not ideal candidates for using this data availability model. Live broadcast streams will likely favor a sliding s_0 data boundary. For implementations that can support live broadcast streams where the s_0 data boundary is fixed (for example, by buffering/recording the data to a local hard disk), the "full random access data availability" model is more appropriate when the buffered content is served without having to go through a content transformation.

See the comment in 7.5.4.2.15 for help with interpreting the "relative to the complete content binary" phrase.

7.5.5 HTTP transport

7.5.5.1 General

There are many possible transport protocols that can be used for the transfer of content. The baseline mandatory media transport protocol for DLNA devices is HTTP. For HTTP the following terms are used:

- HTTP client endpoint – the DLNA entity that issues the HTTP GET or POST request;
- HTTP server endpoint – the DLNA entity that receives the HTTP GET or POST request and issues an HTTP response (possibly including data);
- streaming HTTP (client/server) endpoint – an HTTP client or server endpoint that processes streaming transfers;
- target response: when a client makes a GET request to obtain a certain resource from the server, the server normally responds with a message that includes a representation of the resource as its entity body. This type of response is called here a target response to

differentiate it from other equally valid responses that do not involve transferring the requested resource (for example, redirections, authorization requests, error messages, etc). Similarly, for HEAD requests, a target response is the same response that servers would form to satisfy the matching GET request, but without carrying the resource representation as its entity body.

The generic term "HTTP endpoint" is used to represent either an HTTP client endpoint or an HTTP server endpoint.

It should be noted that the guidelines specified in 7.5.5 apply to DLNA content transactions between DLNA device classes or device capabilities. These guidelines do not specify behaviour for non-DLNA devices. A DLNA device class or device capability may be implemented by software running on a more general-purpose device/platform. For example, the HTTP server of a DMS may be used to serve DLNA and non-DLNA content. These guidelines apply only when the DLNA content is being served to a DLNA device.

Subclause 7.5.5 is organized into the following groups.

- HTTP media transport common guidelines (7.5.5.2.1 to 7.5.5.2.38): This group contains guidelines that are common to all HTTP transfers. These guidelines are independent of the transfer mode.
- HTTP media transport for streaming transfer (7.5.5.3.1 to 7.5.5.3.19): This group contains guidelines that are specific to the use of HTTP for streaming transfers.
- HTTP transport – Interactive transfer (7.5.5.4.1 to 7.5.5.4.3): This group contains guidelines that are specific to the use of HTTP for interactive transfers.
- HTTP transport – Background transfer (7.5.5.5.1 to 7.5.5.5.3): This group contains guidelines that are specific to the use of HTTP for background transfers.
- HTTP transport – POST (7.5.5.6.1 to 7.5.5.6.5): This group contains guidelines that are specific to HTTP POST transactions. HTTP POST transactions always work in conjunction with all other applicable HTTP guidelines.

7.5.5.2 HTTP transport – Common requirements, HTTP media transport common guidelines

7.5.5.2.1 MT baseline transport – HTTP

7.5.5.2.1.1 A DLNA device must implement HTTP as the mandatory media transport with constraints and extensions defined in subsequent entries of 7.5.5.2.1.

7.5.5.2.7 and 7.5.5.2.26 define the HTTP version expectations for HTTP servers and clients.

M	L	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+. +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[39]

DLNA specifies HTTP as the baseline media transport.

7.5.5.2.1.2 DLNA devices must follow the syntax rules for HTTP headers defined in [39] unless DLNA defines syntax for the HTTP header value.

M	R	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+. +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[39]

It should be noted that the BNF rules used in [39] is slightly different from that in DLNA. For example, the default treatment of literals in [39] is case-insensitive. Furthermore, [39] uses implied LWS between tokens and separator.

7.5.5.2.1.3 If the DLNA guidelines define a BNF syntax for an HTTP header, then DLNA devices must not include white spaces in the header-value of HTTP headers unless SP and LWS are explicitly specified in the syntax (BNF) definitions.

If the DLNA guidelines do not define a BNF syntax for an HTTP header, then the header must conform to the message-header syntax in 4.2 of [39], regardless of whether the HTTP header is defined in [39] or if the HTTP header is vendor-defined. It should be noted that the syntax for field-value permits LWS to separate tokens and other data in the field-value.

Implied LWS between the HTTP header-name and the HTTP header-value are allowed as specified in [39], regardless of whether the DLNA guidelines specify a BNF syntax for the HTTP header.

The following cases are allowed examples:

- Range: bytes=1539686400-;
- Content-Range:bytes 21010-47021/47022;
- TimeSeekRange.dlna.org : npt=00:05:35.3-00.

The following examples are not allowed:

- Range: bytes = 1539686400-;
- Content-Range:bytes 21010-47021 / 47022;
- TimeSeekRange.dlna.org : npt = 00 : 05 : 35.3-00.

M	L	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[39]

Reference [39] allows, including white spaces, between any two adjacent words (token or quoted-string), and between adjacent words and separators, for example, "=", SP, "/", ":" (see "implied *LWS" in 2.1 of [39]), but this guideline restricts "implied *LWS" to simplify white space rules for HTTP headers.

It should be noted that it is still allowed to include white spaces between header-name and header-value. The header-name and header-value are defined in 4.2 of [39].

This guideline applies to HTTP headers defined in [39] and other HTTP headers, for example DLNA and vendor defined headers, with the DLNA guidelines. At least one LWS must be inserted between tokens in case of the "*" rule" syntax (for example, USER-AGENT and SERVER headers).

7.5.5.2.2 MT HTTP graceful recovery

HTTP client or server endpoints should not require a hardware reset or a power cycle to return to normal operating conditions after encountering improperly terminated HTTP connections.

S	A	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	n/a

This guideline specifies that a media endpoint needs to be able to handle scenarios where an HTTP connection is not properly terminated. Network conditions and/or HTTP server endpoint behaviour may cause this scenario to occur. Although a full definition for graceful recovery is not provided, a baseline expectation is provided. Essentially, users should not need to reset or power cycle the device simply because a content transfer was interrupted.

7.5.5.2.3 MT HTTP DLNA URI usage

HTTP client endpoints that issue HTTP requests on DLNA URIs (such as those obtained from a UPnP AV ContentDirectory service implementation) may assume that the URI value is properly URI escaped. No additional URI escaping logic is required of a client endpoint.

O	C	DMP +PU+ DMR +DN+ DMP+ +UP+	M-DMS M-DMD M-DMP M-DMU	MIU	[35] [39] [55]

This guideline permits an endpoint to use URI values (obtained from a DLNA endpoint) without having to implement logic for escaping the URI. This guideline is a clarification of 7.4.1.3.10 MM URI rules, which states that DMS devices must advertise URI values that are URI escaped. Although an endpoint can implement additional logic for validating a URI, such logic is useful only for interoperability with non-DLNA devices.

This guideline applies generally, including HTTP GET, HEAD, and POST requests.

7.5.5.2.4 MT valid HTTP response

7.5.5.2.4.1 If an HTTP server endpoint receives an HTTP request, then the endpoint must send a valid HTTP response provided it has sufficient platform resources (network sockets, stored file in readable state, available tuner hardware, etc.) for sending the response.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39] [35]

This guideline essentially obligates an endpoint to send an HTTP response to an HTTP request. It should also be noted that the HTTP specification already obligates a server to return a valid HTTP response for each received HTTP request. Valid HTTP responses include among others: content byte data responses, requests for authorization, HTTP error responses, etc.

7.5.5.2.4.2 If an HTTP server endpoint cannot respond to an HTTP request by sending or receiving content byte data due to the server capacity, network capacity, or current state of the device (such as a tuner locked in a recording state), then the HTTP server should respond with an HTTP error response code of 503 (service unavailable).

S	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

This guideline covers the case where the endpoint has the resources to send an error response but lacks the resources to send or accept content data. Sending an HTTP error is better than denying the TCP connection request because it explicitly tells the requesting endpoint that content cannot be handled at this moment. It should be noted that HTTP servers will respond with other HTTP error codes when responding to other error scenarios, as indicated in the HTTP specification.

7.5.5.2.4.3 If an HTTP server endpoint cannot respond to an HTTP request by sending or receiving content byte data due to the device's lack of available network sockets, then the endpoint may refuse to create new TCP connections for answering content requests.

O	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[20] [39]

This guideline permits an HTTP server to refuse the creation of new TCP connections when it lacks the resources for transporting content. Although this behaviour is allowed by standard convention, endpoints may continue to retry the creation of a TCP connection. Therefore, whenever the situation is both appropriate and possible, HTTP servers are encouraged to respond with an HTTP 503 error.

7.5.5.2.4.4 HTTP client endpoints that issue requests (for example, for playback, download operation, or any normative system usage) must be capable of performing such operations even if the HTTP server endpoint accepts only a single open HTTP connection at any given time.

HTTP client endpoints that claim support for a media operation for a particular content binary must be capable of performing such operations even if the HTTP server endpoint accepts only a single open HTTP connection at any given time.

M	C	DMP DMR DMP _r +DN+ +UP+	M-DMP M-DMD M-DMU	MIU	[39]

Some HTTP server endpoints may accept multiple simultaneous HTTP connections, but others may accept only one. This guideline ensures that HTTP client endpoints provide reliable services even with the most constrained case (only one HTTP connection available).

For example, the following procedures in HTTP client endpoints will not work with HTTP server endpoints that support a single HTTP connection:

- obtaining an IFO file in parallel to a content transfer connection;
- playback transitions between normal speed playback and trick mode playback with multiple HTTP sessions that overlap in time;
- tuner channel changes where one HTTP connection is used for the current channel and another time-overlapping HTTP connection is used for the new channel selection.

7.5.5.2.4.5 HTTP server endpoints should support more than one simultaneous HTTP media transport connection.

S	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

It is a good practice for an HTTP server endpoint to support multiple HTTP client endpoints simultaneously.

7.5.5.2.4.6 If an HTTP server endpoint has not completed the transmission of an HTTP response and the HTTP client endpoint wants to stop the current data flow to issue a new request, then the HTTP client endpoint should close the existing TCP connection and then create a new TCP connection for the new HTTP request.

S	C	DMP DMR DMPr +DN+	M-DMP M-DMD	MIU	[39]

Since clients cannot assume that endpoints can support multiple HTTP connections, this guideline recommends that clients use one TCP connection. Implementers of HTTP servers and clients should also consider this guideline in conjunction with guidelines 7.5.5.3.12.1 and 7.5.5.3.12.2, which deal with scan operation playback (a.k.a. trick-modes) for streaming transfers.

7.5.5.2.4.7 An HTTP server endpoint must begin sending a response message to an HTTP requests within 27 s of receiving the request. Valid response messages must be either the response with content byte data, or appropriate error messages.

M	L	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

This guideline defines the maximum response time for an HTTP server endpoint for a request for content.

In conjunction with 7.5.5.2.4.9, these guidelines ensure HTTP transactions for media transport.

It should be noted that the time-out value is for the worst case. It is recommended that an HTTP server endpoint responds to a request as soon as possible.

7.5.5.2.4.8 An HTTP client endpoint must wait at least 30 s before closing the TCP connection if it has not received any response from the HTTP server endpoint to an HTTP GET request for content.

M	L	DMP DMR DMPr +DN+	M-DMP M-DMD	MIU	[39]

This guideline is not subject to scenarios involving user cancellation. A connection can be cancelled through user intervention at any time.

This does not imply that the entire content binary will be received within the 30 s, only that it has started.

7.5.5.2.4.9 HTTP server endpoints must be capable of providing at least one media transport HTTP connection and must also be capable of processing sequential HTTP requests without responding with an HTTP error response code 503 (service unavailable).

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

This guideline ensures that rendering endpoints are capable of rendering a content binary using one media transport HTTP connection (see 7.5.5.2.4.4).

This guideline is not subject to preventing content sources to respond with 503 (service unavailable) when it currently is unable to process a HTTP request (see 7.5.5.2.4.2), but a content source must never return 503 (service not available) under "test conditions".

7.5.5.2.4.10 An HTTP server endpoint must use the HTTP status code of 503 (service unavailable) for an HTTP request only on the conditions that the content source does not

have sufficient platform resources (network sockets, stored file in readable state, available tuner hardware, etc.) for sending the response with content byte data. On other conditions, the HTTP server endpoint must not use the HTTP status code of 503 (service unavailable) for an HTTP request.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

This guideline defines the maximum response time for an HTTP server endpoint for a request for content.

In conjunction with 7.5.5.2.4.9, these guidelines ensure HTTP transactions for media transport.

It should be noted that the time-out value is for the worst case. It is recommended that an HTTP server endpoint responds to a request as soon as possible.

7.5.5.2.4.11 An HTTP client may treat HTTP status code 503 (service unavailable) as equivalent to HTTP status code 500 (internal server error)

O	C	DMP DMR DMP+DN+ +UP+	M-DMP M-DMD M-DMU	MIU	[39]

The guideline clarifies a content receiver behaviour for response message 503 with or without Retry-After. Hence, no retry is required.

7.5.5.2.5 MT HTTP header tolerance

7.5.5.2.5.1 HTTP client and server endpoints must be tolerant of unknown HTTP headers.

Tolerant behaviour is defined as being able to successfully "parse and interpret" or "parse and ignore" the HTTP headers and their values.

M	R	DMS DMP DMR DMP+DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M-DMD M-DMU	MIU	[27] [39]

This guideline addresses forward compatibility and allows for broader interoperability with implementations that employ transport layer vendor extensions by way of HTTP headers.

7.5.5.2.5.2 Each HTTP header line (including the header's name and value but excluding the last carriage-return/line-feed sequence, CRLF) must not exceed 998 bytes.

M	R	DMS DMP DMR DMP+DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M-DMD M-DMU	MIU	[39] [40]

These guidelines limit the length of an HTTP header line according to 2.2.1 of [40]. The guidelines also specify the normative way to encode HTTP headers that span multiple lines.

Multi-line HTTP headers are always split at LWS characters.

7.5.5.2.5.3 If an HTTP header line (header's name and value but excluding the last CRLF) exceeds 998 bytes, then the HTTP header must span multiple lines. HTTP headers that span multiple lines must prefix the additional lines with at least one space (SP) or horizontal tab (HT) as described in 4.2 of [39].

M	R	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[39] [40]

7.5.5.2.6 MT HTTP header case-sensitivity

Names of HTTP headers must be treated as case insensitive tokens.

M	R	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[27] [39]

This is normative according to the HTTP specification.

7.5.5.2.7 MT baseline transport – HTTP server endpoints

7.5.5.2.7.1 HTTP server endpoints used for media transport purposes must be compliant with HTTP/1.1, which also requires HTTP/1.0 compliance.

M	L	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

HTTP servers need to support both HTTP/1.0 and HTTP/1.1 requests to ensure wide interoperability.

7.5.5.2.7.2 HTTP/1.1 server endpoints used for media transport should return HTTP version 1.1 in the response header, regardless of the version specified in the HTTP client's request.

S	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[29]

The clarifying RFC [29] clarifies that HTTP/1.1 servers should return HTTP/1.1 even if the HTTP server receives a request marked with HTTP/1.0. The robustness rules, specified by the HTTP specification, enables clients and servers that employ different HTTP version numbers to coexist properly.

It should be noted that message format refers to both the HTTP headers and HTTP response body. As described by [29], the version field in a response message header indicates the protocol level that the server is capable of understanding. However, a server that understands protocol version 1.1 may generate messages compatible with version 1.0 in order to communicate with clients capable of handling only the lower 1.0 version. When this happens, the response message has a version header equal to 1.1 but the format of the message contains only version 1.0 headers. Reference [29] provides more details for interoperability between hosts with different HTTP versions.

7.5.5.2.7.3 When responding to a request of version 1.0, an HTTP server endpoint must format the response message in such a way that the result of decoding and processing the message does not depend on headers outside the scope of the HTTP 1.0 specification.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[29] [39]

7.5.5.2.7.4 HTTP server endpoints must not report a higher version of HTTP than is actually supported by the implementation.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.7.5 Interoperability between HTTP client and server endpoints that implement different versions of the HTTP protocol must follow the provisions and recommended actions defined in [29].

M	R	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M-DMD M-DMU	MIU	[29]

Reference [29] defines the significance of HTTP version numbers, the rules for interoperability between hosts with different version numbers, and rules for the actual version number that should be included when creating messages.

7.5.5.2.7.6 HTTP server endpoints should support persistent HTTP/1.1 connections and pipelined HTTP/1.1 requests.

S	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

The default behaviour for HTTP servers responding to HTTP/1.1 requests is to support persistent connections, which means that the HTTP server can respond to multiple HTTP/1.1 requests on one HTTP session. Pipelined requests may be used to facilitate seek operations. See 7.5.5.6.5 for more information.

7.5.5.2.8 MT HTTP header – scmsFlag.dlna.org

7.5.5.2.8.1 HTTP client and server endpoints may use the scmsFlag.dlna.org HTTP header, which indicates copyright assertion and copy status flags when transporting audio only content. HTTP server endpoints that serve DLNA media content and non-DLNA media content may also use this flag for the latter. HTTP client endpoints that encounter this HTTP header may implement behaviour to enforce regional copyright provisions.

O	A	DMS DMP +PU+ +UP+ +DN+	M-DMS M-DMP M-DMD M-DMU	MIU	[39] [72]

These guidelines make it possible to comply with regional legal requirements, such as in [72]. The flag is to be used with both DLNA and non-DLNA audio only content. The syntax of the value is strictly defined by these guidelines.

7.5.5.2.8.2 The notation of scmsFlag.dlna.org header field is defined as follows.

- scmsFlag.dlna.org = "scmsFlag.dlna.org" *LWS ":" *LWS flagValue
- flagValue = "00" | "01" | "10" | "11"

The value of the scmsFlag.dlna.org header must be a two letter string from the following list: "00", "01", "10" or "11". The first and second characters can be set to 0 or 1 independently according to the rules below:

Definition of the value of the 1st character (i.e., leftmost) of the scmsFlag.dlna.org HTTP header:

- 0: copyright is asserted
- 1: no copyright is asserted

Definition of the value of the 2nd character of scmsFlag.dlna.org HTTP header:

- 0: Original recording
- 1: First generation or higher recording

The following example means copyright is asserted and first-generation or higher recording.

- scmsFlag.dlna.org: 01

M	A	DMS +PU+ +UP+	M-DMS M-DMU	MIU	[39] [72]

7.5.5.2.9 MT HTTP header – Content-Type

7.5.5.2.9.1 HTTP server endpoints must specify the Content-Type HTTP header in the HTTP response header fields whenever it returns a target response to an HTTP GET operation.

M	L	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

It is imperative that endpoints specify the Content-Type field to allow the receiver to know the MIME TYPE for the content that is to be sent in the HTTP response message.

It should be noted that MIME-TYPE values appear in the Content-Type HTTP header.

7.5.5.2.9.2 The MIME-TYPE values that appear in the compendium of media format profiles, Clause 5 in IEC 62481-2, must be used as values for Content-Type when an HTTP message describes DLNA media contents.

M	L	DMS +PU+ +PR1+ +PR2+ +UP+	M-DMS M-DMU	MIU	IEC 62481-2

This guideline specifies the correct mime type values for use with the Content-Type header field when transporting content encoded in a DLNA media format.

7.5.5.2.9.3 HTTP client endpoints must specify the Content-Type HTTP header in the HTTP request if using a POST operation to send data.

M	L	+UP+	M-DMU	MIU	[39]

7.5.5.2.10 MT HTTP Header – contentFeatures.dlna.org

7.5.5.2.10.1 If an HTTP server endpoint receives an HTTP GET or HEAD request with the getContentFeatures.dlna.org HTTP header, then the HTTP server must use the contentFeatures.dlna.org HTTP header if it responds with a target response.

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39] [55]

As noted, this guideline permits an HTTP server endpoint to use the contentFeatures.dlna.org in an HTTP response.

7.5.5.2.10.2 An HTTP server endpoint may respond with the contentFeatures.dlna.org HTTP header to an HTTP GET or HEAD request that does not have the getContentFeatures.dlna.org HTTP header.

O	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39] [55]

7.5.5.2.10.3 HTTP client endpoints may use the getContentFeatures.dlna.org when issuing GET or HEAD requests.

O	A	DMP DMR DMPr +DN+ +UP+	M-DMP M-DMD M-DMU	MIU	[39] [55]

These guidelines describe how an HTTP client endpoint can request an HTTP server to use the contentFeatures.dlna.org in the response.

7.5.5.2.10.4 The notation of getContentFeatures.dlna.org header field is defined as follows:

- getContentFeatures.dlna.org = "getContentFeatures.dlna.org" *LWS ":" *LWS "1"

The only value possible is "1".

Example:

- getContentFeatures.dlna.org: 1

M	A	DMS DMP DMR DMPr +DN+ +UP+ +PU+	M-DMS M-DMP M-DMD M-DMU	MIU	[39] [55]

7.5.5.2.10.5 If an HTTP server endpoint receives any value except "1" in the getContentFeatures.dlna.org header, it must return an error code response of 400 (bad request).

M	A	DMS +PU+	M-DMS	n/a	[39] [55]

7.5.5.2.10.6 The value of the contentFeatures.dlna.org HTTP header must be the same value as the 4th field of the content's res@protocolInfo value, as described in the 7.4.1.3.18 MM opparm (operations parameter common guidelines) guideline.

The notation of contentFeatures.dlna.org header field for DLNA media transport is defined as follows:

- contentFeatures-line = "contentFeatures.dlna.org" *LWS ":" *LWS 4th-field;
- 4th-field = <case sensitive 4th field value defined in guideline 7.4.1.3.18.1>

M	A	DMS +PU+ +UP+ +PR1+ +PR2+	M-DMS M-DMU	MIU	[39] [55]

This guideline allows HTTP transport transactions to carry information (that is normally only accessible at the UPnP AV ContentDirectory service layer) about the requested content (and the server capabilities for that content).

It should be noted that this header may be used by content sources for non-DLNA content.

7.5.5.2.10.7 HTTP client endpoints must not include the following headers in HTTP requests, unless the appropriate protocolInfo 4th field parameters indicate support for them:

- Range;
- TimeSeekRange.dlna.org;
- PlaySpeed.dlna.org;
- getAvailableSeekRange.dlna.org.

M	A	DMP DMR DMP+ +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39] [55]

The 4th field of a protocolInfo is obtained from contentFeatures.dlna.org. Devices can also directly check metadata in the CDS, without using contentFeatures.dlna.org.

These guidelines do not define interoperability guidelines for a scenario where an HTTP client endpoint attempts to use an optional transport layer feature when the 4th field does not indicate support for the transport layer feature.

7.5.5.2.10.8 An HTTP client that issues a POST request for uploading DLNA-conformant content must use the contentFeatures.dlna.org HTTP header. The value sent must match the 4th field that was sent in the CDS:CreateObject request, as defined in 7.4.1.7.19.6.

M	A	+UP+	M-DMU	MIU	[39] [55]

These guidelines explain how contentFeatures.dlna.org is used in POST transactions.

7.5.5.2.10.9 An HTTP server that receives a POST request

- without the contentFeatures.dlna.org HTTP header or

- with the incorrect DLNA.ORG_PN parameter

then the HTTP server may respond with an HTTP error code 406.

O	C	DMS	M-DMS	n/a	[39] [55]

7.5.5.2.11 MT HTTP Header – dlina pragma-directive (ifoFileURI.dlna.org)

7.5.5.2.11.1 If an HTTP server endpoint provides a res@dlina:ifoFileURI for a resource (see 7.4.1.4.8) and the HTTP server endpoint receives an HTTP GET or HEAD request with the Pragma-with-getIfoFileURI-pragma-directive in the HTTP request, then the HTTP server endpoint must provide the Pragma-with-ifoFileURI-pragma-directive (with the res@dlina:ifoFileURI value associated with the target URI) if it responds with a target response.

If an HTTP server endpoint does not provide a res@dlina:ifoFileURI for a resource, then the HTTP server must not provide the Pragma-with-ifoFileURI-pragma-directive in the HTTP response.

M	A	DMS +PU+	M-DMS	n/a	[39] [55]

Section 14.32 of [39] defines the pragma as a general header for implementation-specific directives.

The DLNA.ORG_PN parameter in the contentFeatures.dlna.org header field indicates the DLNA media format profile ID. Furthermore, as described in 7.4.1.4.8, content profiled as MPEG_PS_NTSC or MPEG_PS_PAL may have an IFO file. If the HTTP response to a request with the Pragma-with-getIfoFileURI-pragma-directive does not include the Pragma-with-ifoFileURI-pragma-directive, then the HTTP client endpoint should be aware that the AV content does not have an IFO file.

Implementers must be careful not to exceed the maximum 998 byte limit for HTTP header lines when using the Pragma-with-ifoFileURI-pragma-directive.

7.5.5.2.11.2 HTTP client endpoints may use the Pragma-with-getIfoFileURI-pragma-directive when issuing GET or HEAD requests.

O	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[39] [55]

7.5.5.2.11.3 The notation of the PRAGMA header field with the getIfoFileURI-pragma-directive is defined as follows:

- Pragma-with-getIfoFileURI-pragma-directive = "PRAGMA" *LWS ":" *LWS *(pragma-directive *LWS "," *LWS) getIfoFileURI-pragma-directive *(*LWS "," *LWS pragma-directive);
- pragma-directive = "no-cache" | extension-pragma;
- extension-pragma = token ["=" (token | quoted-string)];
- getIfoFileURI-pragma-directive = "getIfoFileURI.dlna.org".

It should be noted that the PRAGMA header name, pragma-directive token, and the getIfoFileURI-pragma-directive token are case insensitive.

Examples:

- PRAGMA: getIfoFileURI.dlna.org

- PRAGMA: getIfoFileURI.dlna.org, no-cache

M	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[39] [55]

The "1#token" syntax means a comma separated list including one or more elements. It should be noted that LWS is allowed before and after the separator comma ',' in this syntax. See 2.1 of [39].

It should be noted that the Pragma-with-getIfoFileURI-pragma-directive may be used by endpoints issuing an HTTP request on non-DLNA content.

7.5.5.2.11.4 The notation of the PRAGMA header field with the ifoFileURI-pragma-directive is defined as follows:

- Pragma-with-ifoFileURI-pragma-directive = "PRAGMA" *LWS ":" *LWS *(pragma-directive *LWS "," *LWS) ifoFileURI-pragma-directive *(*LWS "," *LWS pragma-directive);
- pragma-directive = "no-cache" | extension-pragma;
- extension-pragma = token ["=" (token | quoted-string)];
- ifoFileURI-pragma-directive = "ifoFileURI.dlna.org" "=" quoted-absolute-uri-string;
- quoted-absolute-uri-string = <same value as the corresponding res@dlna:ifoFileURI attribute value, as described in 7.4.1.4.8 MM IFO File. It must be quoted by "">.

Example:

- PRAGMA: ifoFileURI.dlna.org="http://192.168.0.1:8080/IFO_101.ifo"

The PRAGMA HTTP header name, the pragma-directive token, and the ifoFileURI-pragma-directive token are case-insensitive.

A content source that provides the Pragma-with-ifoFileURI-pragma-directive with non-DLNA content must also provide an IFO file that conforms to the guidelines in MPEG2 AV Format, IFO File Format, Profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.19 in IEC 62481-2.

M	A	DMS +PU+	M-DMS	n/a	[39] [55], IEC 62481-2

7.5.5.2.12 MT HTTP HEAD requests

7.5.5.2.12.1 HTTP server endpoints (HTTP/1.1 and HTTP/1.0) must respond to HTTP HEAD requests, using the guidelines described below.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

There are several interpretations to the format of HEAD responses. These guidelines provide consistent interpretation.

7.5.5.2.12.2 A target response to a HEAD request must be composed of only HTTP headers and a zero-length response entity body.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.12.3 Target responses to identical HTTP version HEAD and GET requests for a given content binary must use consistent transfer encoding headers. For example, if Content-Length is included in a HEAD target response then Content-Length must also be present in the GET target response for the same content binary.

This guideline assumes that the content binary has not changed between the different HTTP requests.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

This guideline forbids HTTP server endpoints from responding to a GET request using a different encoding method than the HEAD target response. For example A DMS could not respond to a HEAD request with Content-Length and respond to a GET request using chunked encoding.

7.5.5.2.12.4 If an HTTP server does not know the length of a requested resource, such as in the case when chunked transfer coding is employed in HTTP/1.1, the Content-Length field must be omitted from the HEAD target response.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.12.5 If an HTTP server endpoint (HTTP/1.1 and HTTP/1.0) responds to an HTTP GET request with a non-error response, the HTTP server should respond to the equivalent HEAD request with a non-error response.

A successful response is defined as an HTTP response with a status code in the 1xx or 2xx range.

S	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

HTTP specification requires HTTP servers to respond to HEAD requests. This guideline clarifies that HTTP servers cannot respond with an error code for HEAD requests that target a valid URI for the HTTP server.

This is not mandatory because conditions may be different from those of the GET request (for example, server saturation, etc.).

7.5.5.2.12.6 The HTTP headers of a HEAD target response must include the Content-Type HTTP header.

M	L	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

Ideally, an HTTP server can know all of the HTTP headers (for requests that use TimeSeekRange.dlna.org, Range, PlaySpeed.dlna.org, or other HTTP headers) that will be sent in an HTTP response without doing any of the computational work to buffer content data, but some scenarios (such as those involving transcoding, live streams, random access requests, etc.) require a lot of computational cycles. For these reasons, these guidelines specify minimal expectations for HEAD responses while recommending the ideal expectations.

7.5.5.2.12.7 HTTP HEAD target responses should have the exact same HTTP headers as those in the equivalent HTTP GET target response.

S	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.12.8 An HTTP client should not issue an HTTP HEAD request to a URI intended for a file upload (for example, res@importUri, res@dna:importIfoFileURI).

S	C	+UP+	M-DMU	MIU	[39]

The MediaServers behaviour of a HEAD request is undefined because implementations that use the same URI value for res@importUri and the <res> value might choose to only respond in the manner of a GET request.

7.5.5.2.13 MT image file size acquisition via HTTP HEAD

7.5.5.2.13.1 An HTTP server that serves an image content binary should respond to HTTP/1.0 HEAD requests with the Content-Length header to indicate the length of the image content binary.

S	L	DMS +PU+ +PR1+	M-DMS	n/a	[39]

HTTP/1.0 HEAD requests allow control points to query a server for the length of an image file, when res@size metadata is not available.

The reason why HTTP/1.0 is used for this purpose is that chunked transfer coding and Content-Length are used in a mutually exclusive manner. Since HTTP servers are required to support HTTP/1.1, many servers use chunked transfer coding. Furthermore, the interoperability guidelines require HTTP headers to be the same for equivalent GET and HEAD requests. Therefore, using HTTP/1.0 for the HEAD request improves the odds of success.

It should be noted that control points that fail to acquire the file size in this manner are still governed by 0 (MM XHTML-Print), meaning that a control point that accidentally specifies a page that exceeds the permitted image byte total is in violation of the guidelines.

7.5.5.2.13.2 A printing control point that attempts to acquire the length of an image content binary by using an HTTP HEAD request should use an HTTP/1.0 HEAD request.

S	L	+PR2+	n/a	n/a	[39]

7.5.5.2.14 MT HTTP header parsing (server)

HTTP server endpoints must gracefully skip over unsupported HTTP header fields. Under no circumstances can an HTTP server fail to process a properly formatted HTTP request because of an unrecognized or unsupported HTTP header field.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

Incorrect HTTP header parsing has been the source of numerous compatibility issues during plugfest events.

7.5.5.2.15 MT HTTP header – Content-Length

7.5.5.2.15.1 If the Content-Length HTTP header is present in an HTTP message with a message body, then the message must not use chunked transfer coding.

M	C	DMS +PU+ +UP+	M-DMS M-DMU	MIU	[39]

The use of Content-Length is not allowed under 4.4 of the HTTP specification in [39].

Applies to both HTTP GET response messages and POST request messages.

7.5.5.2.15.2 If the Content-Length HTTP header is omitted from an HTTP GET response, then the HTTP server endpoint must do one of the following.

- The HTTP server will close the TCP connection immediately after sending the last byte of the response message. Furthermore, if the HTTP server is responding to an HTTP/1.1 transaction, then the HTTP server must also use the CONNECTION: CLOSE header and value to explicitly indicate that it will close the connection. Lastly, any additional byte sequence following the headers must be treated as entity-body bytes until the instant when the connection is closed.
- The HTTP/1.1 server will use chunked transfer coding for the response when communicating with an HTTP/1.1 client.

This guideline applies in all scenarios with the following exceptions:

- response messages that are prohibited from having an entity-body (such as the 1xx, 204, and 304 responses);
- the HTTP server returns an HTTP/1.1 response with no entity-body, chunked transfer coding is not used, and the CONNECTION:CLOSE header is not used (i.e., persistent connection).

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

These guidelines clarify the expected behaviour regarding Content-Length use for response messages.

For pipelined requests, if the server decides to close a TCP connection for some response, any additional requests submitted afterwards will not be processed by the server.

If the Content-Length is used in messages with no entity body, then the accurate value of "0" is required according to guideline 7.5.5.2.15.5. Likewise, messages encoded with chunked transfer coding will use the accurate value of "0" for the chunk-size.

7.5.5.2.15.3 If the Content-Length HTTP header is omitted from an HTTP POST request, then the HTTP client endpoint must use chunked transfer coding for the request.

M	C	+UP+	M-DMU	MIU	[39]

See 4.4 of [39]

7.5.5.2.15.4 An HTTP message that carries no content binary data in the entity body, and which uses chunked transfer coding must use a single chunk with chunk-size indicator equal to 0.

M	R	DMS +PU+ +UP+	M-DMS M-DMU	MIU	[39]

Some response messages do not carry content binary data in the message payload. If the HTTP server endpoint is using chunked transfer coding to produce this message, then the message has a single one-line chunk-size indicator with a value of 0.

7.5.5.2.15.5 When operating under persistent connections (including pipelining), an HTTP/1.1 client must detect the existence of a message entity body when it receives a message with

- a non-zero Content-Length header;
- non-zero chunk-size values when using chunked transfer coding;
- non-zero content bytes following the message headers when the HTTP server declares that it will close the connection. (An HTTP server declares that it will close the connection by using the "CONNECTION: CLOSE" header in the response.)

If the Content-Length header and the CONNECTION:CLOSE header are not provided and chunked transfer coding is not used in the HTTP/1.1 response, then the message has no entity body.

M	C	DMP DMR DMP+ +DN+	M-DMP M-DMD	MIU	[39]

This guideline clarifies the process used by a client to parse and extract the body (if any) of received response messages.

It should be noted that response messages that do not carry a Content-Length and do not use transfer encoding, could carry content bytes if the server closes the connection after sending the last byte. The server needs to explicitly announce that the connection will be closed by using the adequate header.

7.5.5.2.15.6 When operating under persistent connections (including pipelining), an HTTP/1.1 server must detect the existence of a message entity body when it receives a POST message with

- a non-zero Content-Length header;
- non-zero chunk-size values when using chunked transfer coding.

M	C	DMS	M-DMS	n/a	[39]

This guideline clarifies the process used by a server to parse and extract the body (if any) of a request message.

7.5.5.2.15.7 If an HTTP server endpoint knows the byte length of a response body, then the HTTP server should use the Content-Length HTTP header in the HTTP target response.

S	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

As a general rule, the Content-Length provides useful information to HTTP clients. However, the Content-Length HTTP header is not required because it is difficult to provide an accurate byte length in some scenarios. For example, HTTP servers that are transmitting live content (and some transcoded content) may not know the value for the Content-Length HTTP header field. In cases when the Content-Length is provided, the value needs to match the byte length of the response body.

7.5.5.2.15.8 If the HTTP server endpoint does not know the byte length of the response entity body or if Content-Length cannot be used due to some exceptions listed in 4.4 of [39], then HTTP servers must omit the Content-Length HTTP header from HEAD, and GET target responses.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.15.9 If an HTTP server endpoint sends an HTTP GET target response with the Content-Length HTTP header, then the byte length of the response entity body must match the value of the Content-Length HTTP header.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.15.10 If an HTTP server endpoint receives an HTTP/1.0 GET request and sends an HTTP target response that does not have a Content-Length HTTP header, then the HTTP server must close the TCP connection when all data is transferred.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

The HTTP server is required to close the TCP connection in this scenario because only the HTTP server knows when it has finished sending the bytes for the requested URI.

7.5.5.2.15.11 If an HTTP client endpoint knows the byte length of the entity-body in a POST request, then the HTTP client should use the Content-Length HTTP header in the HTTP request.

S	R	+UP+	M-DMU	MIU	[39]

7.5.5.2.15.12 If the HTTP client endpoint does not know the byte length of the entity-body in a POST request or if Content-Length cannot be used, the HTTP client endpoint must omit the Content-Length HTTP header from the POST request and use the chunked transfer coding as specified in 4.4 of [39].

M	R	+UP+	M-DMU	MIU	[39]

7.5.5.2.15.13 If an HTTP Client Endpoint sends an HTTP POST request with the Content-Length HTTP header, then the byte length of the entity-body must match the value of the Content-Length HTTP header.

M	R	+UP+	M-DMU	MIU	[39]

7.5.5.2.16 MT maximum byte size transfers

7.5.5.2.16.1 HTTP client and server endpoints must not use values that exceed 281474976710655 (i.e., $2^{48} - 1$) for the following HTTP fields.

- Content-Length header;
- first byte pos and last byte pos (as defined in 14.35.1 and 14.16 of [39] and guideline 7.5.5.2.22.4);
- instance-length (as defined in 14.16 of [39]);
- chunk-size (as defined in 3.6.1 of [39]).

M	L	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[39]

The HTTP specification [39] does not limit the maximum content length. It should be noted that a 32-bit integer is not sufficient, especially, for a 2 h MPEG-2 stream that exceeds 4 GBytes. The specified range covers the maximum size of a DLNA content binary.

It should be noted that chunk-size is in hexadecimal form, while the other fields are in decimal form.

7.5.5.2.16.2 An HTTP client or server endpoint must parse and interpret values up to 281474976710655 (i.e., $2^{48} - 1$) for the Content-Length header field, range units in bytes, and chunk-size field that are represented in HTTP requests/responses.

M	L	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[39]

7.5.5.2.17 MT HTTP/1.0 persistent connections (server)

If an HTTP server endpoint receives an HTTP/1.0 GET or HEAD request with the CONNECTION: KEEP-ALIVE header, then the HTTP server must ignore this header and close the TCP connection after the response is sent.

M	L	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[27] [39]

Although HTTP/1.0 supports persistent connections by way of the KeepAlive token, there are interoperability issues because the methodology is an experimental extension.

7.5.5.2.18 MT HTTP common random access data availability requirements

7.5.5.2.18.1 If an HTTP server endpoint supports the range header field or TimeSeekRange.dlna.org header fields for a content binary, it should support and process an

HTTP HEAD request to get the current random access data range for the content binary while it is processing the HTTP GET request to transmit the target response.

The process for acquiring the random access data ranges for "full random access data availability" and "limited random access data availability" model are described in 7.5.5.2.19 and 7.5.5.2.20 respectively.

All guidelines in 7.5.5.2.18 MT HTTP common random access data availability apply for both "full random access data availability" and "limited random access data availability" models.

S	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

Subclause 7.5.5.2.4.4 states that HTTP clients must not assume that HTTP servers accept more than one media transport HTTP connection at a time. Although an HTTP client must not assume that the HTTP server accepts more than one HTTP GET request simultaneously, this guideline clarifies the intent for HEAD requests. In cases where a random access data range is updated while an HTTP client is receiving streaming data from a HTTP GET request, it is useful for the HTTP client to use an HTTP HEAD request to get the current random access data range simultaneously.

7.5.5.2.18.2 If an HTTP server endpoint supports the TimeSeekRange.dlna.org header field for the specified URI, it must process any requested range in the random access data range at the time.

M	A	DMS +PU+	M-DMS	n/a	[39]

This is a general requirement that random access requests be supported on the entire $[r_0, r_N]$ data range.

7.5.5.2.18.3 If an HTTP server endpoint supports the range header field for the specified URI, it must process any requested range in the random access data range at the time.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

This is a general requirement that random access requests be supported on the entire $[r_0, r_N]$ data range.

7.5.5.2.19 MT HTTP – Data range of "full random access data availability"

7.5.5.2.19.1 When using the "full random access data availability" model, the npt value of 0 and the byte position of 0 for the TimeSeekRange.dlna.org header field must refer to the beginning of the content binary (i.e., the first byte available for the content binary).

All guidelines in 7.5.5.2.19 apply only in the case of "full random access data availability" models.

M	A	DMS DMP DMR +PU+ +DN+	M-DMS M-DMP M-DMD	MIU	[39]

These guidelines clarify 7.5.4.2.15 by explaining how npt values and byte indices are used with the TimeSeekRange.dlna.org and range headers.

7.5.5.2.19.2 If the end position is not specified in a GET request with range or TimeSeekRange.dlna.org, then the HTTP server endpoint's transmission of the target

response must include the content data that is currently available and the content data that will be available in the future for the current stream. (i.e., assuming s_N is changing with time, the server keeps transmitting data until s_N becomes permanently fixed).

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.19.3 The byte position of 0 for the range header field must refer to the beginning of the content binary.

M	A	DMS DMP DMR +PU+ +DN+	M-DMS M-DMP M-DMD MIU	[39]

7.5.5.2.19.4 If an HTTP server endpoint receives an HTTP GET request that omits the range and TimeSeekRange.dlna.org HTTP headers, then the HTTP server endpoint must infer a byte-pos of 0 (i.e., respond from the beginning of the content binary).

This guideline must also apply when s_0 -increasing is false and neither range nor TimeSeekRange.dlna.org are supported by the HTTP server.

M	A	DMS	M-DMS	MIU	[39]

This guideline ensures that HTTP clients will receive content in a manner consistent with the conventions established by HTTP.

7.5.5.2.19.5 If an HTTP server endpoint supports the range HTTP header (as defined in the 7.5.5.2.22) with the "full random access data availability" model, the HTTP server endpoint should specify the instance-length in the Content-Range HTTP header field of the target response to a HTTP GET/HEAD request with the range HTTP header field.

S	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

This guideline clarifies 14.16 of [39]. The asterisk "*" may be used instead of the instance-length if the instance-length is unknown at the time when the response was generated or when s_N -increasing is true. However, DLNA recommends that HTTP servers provide an instance-length.

7.5.5.2.19.6 If an HTTP client endpoint wants to get the instance-length of a content binary, then it should use an HTTP HEAD request with the range HTTP header that specifies 0 for the first-byte-position and omits the end byte-position.

S	C	DMP DMR DMPr +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

The response will include the Content-Range header. If the instance length is known, then it will be the number of bytes in the UCDAM [s_0 , s_N] data range. Otherwise, the instance-length token will be "*".

7.5.5.2.19.7 If an HTTP server endpoint knows the instance-length of a content binary, then it must provide the instance-length in the Content-Range header.

NOTE This guideline is mandatory for scenarios where the HTTP server is serving stored content.

M	A	DMS +UP+	M-DMS	n/a	[39]

7.5.5.2.19.8 If an HTTP server endpoint supports the TimeSeekRange.dlna.org HTTP header (as defined in guideline 7.5.5.2.24) with the "full random access data availability" model, the HTTP server endpoint should specify the instance-duration in the TimeSeekRange.dlna.org HTTP header field of the target response to an HTTP GET/HEAD request with the TimeSeekRange.dlna.org HTTP header field.

S	C	DMS +PU+	M-DMS	n/a	[39]

This guideline clarifies guideline 7.5.5.2.24.3. The syntax does not require an instance-duration, but DLNA recommends specifying the instance-duration.

7.5.5.2.19.9 In conjunction with the 7.5.5.2.19.8, if an HTTP server endpoint supports the TimeSeekRange.dlna.org HTTP header for a content binary, it should specify the instance-length in the TimeSeekRange.dlna.org HTTP header field of the target response to an HTTP GET/HEAD request with the TimeSeekRange.dlna.org HTTP header field.

S	C	DMS +PU+	M-DMS	n/a	[39]

Even if TimeSeekRange.dlna.org is supported without the range header, then the HTTP server should provide an instance-length when possible.

It should be noted that 7.5.5.2.24.5 requires instance-length if the range HTTP header field is also supported.

7.5.5.2.20 MT HTTP – Data range of "limited random access data availability"

7.5.5.2.20.1 If the end position is not specified in a GET request with range or TimeSeekRange.dlna.org, then the HTTP server endpoint's transmission of the target response must include the content data that is currently available and the content data that will be available in the future for the current stream. (i.e., assuming s_N is changing with time, the server keeps transmitting data until s_N becomes permanently fixed.)

All guidelines in 7.5.5.2.20 apply only in the case of "limited random access data availability" models.

M	C	DMS +PU+	M-DMS	n/a	[39]

In the case of live content, the end of the content binary is undefined. If the end position is not specified in the request, it means the client wants to continue receiving data until the absolute end of the stream.

7.5.5.2.20.2 Since the random access data range of $[r_0, r_N]$ is able to change at any time, the HTTP client endpoint should not assume that a GET request (with either range or TimeSeekRange.dlna.org) that specifies a range with a previous random access data range will always succeed.

S	C	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

An HTTP request with a range request may fail with an HTTP response error code of: 416 (requested range not satisfiable) due to changing random access data range.

7.5.5.2.20.3 If the HTTP client endpoint specifies both the TimeSeekRange.dlna.org HTTP header field and the PlaySpeed.dlna.org HTTP header field in an HTTP GET request for a content binary in the case of the "limited random access data availability" model, it must specify the end position of the request range in the range specifier of the TimeSeekRange.dlna.org HTTP header.

M	C	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

Even if the HTTP server supports the PlaySeed.dlna.org HTTP header field for server side's trick mode, it cannot process it beyond the current available data in the case of "limited random access data availability".

7.5.5.2.20.4 If an HTTP server endpoint supports either the range HTTP header (as defined in the guideline 7.5.5.2.22) or the TimeSeekRange.dlna.org HTTP header (as defined in the guideline 7.5.5.2.24) with the "limited random access data availability" model for a content binary, the HTTP server endpoint must support the getAvailableSeekRange.dlna.org HTTP header field and availableSeekRange.dlna.org HTTP header field in HTTP GET/HEAD requests for the content binary.

M	A	DMS +PU+	M-DMS	n/a	[39]

Essentially, a server that supports the "limited random access data availability" model for HTTP needs to implement support for the getAvailableSeekRange.dlna.org and availableSeekRange.dlna.org HTTP headers. These headers allow clients to request the random access data range and allow servers to respond with that information.

7.5.5.2.20.5 If an HTTP server endpoint that supports the availableSeekRange.dlna.org HTTP header field (for a content binary) receives an HTTP HEAD/GET request with the getAvailableSeekRange.dlna.org HTTP header, it must respond with the availableSeekRange.dlna.org HTTP header field to return the current random access data range.

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.20.6 The notation of getAvailableSeekRange.dlna.org header field must be as follows:

- getAvailableSeekRange-line = "getAvailableSeekRange.dlna.org" *LWS ":" *LWS "1"

The only value possible is "1". Any other values sent must result in the HTTP server responding with an error code of 400 (bad request).

Example:

- getAvailableSeekRange.dlna.org: 1

M	A	DMS DMP DMR +DN+ +PU+	M-DMS M-DMP M-DMD	MIU	[39]

7.5.5.2.20.7 If an HTTP server endpoint receives any value except "1" in the `getAvailableSeekRange.dlna.org` header it must return an error code response of 400 (bad request).

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.20.8 The notation of the `availableSeekRange.dlna.org` header field for DLNA media transport must be as follows:

- `availableSeekRange-line` = "availableSeekRange.dlna.org" *LWS ":" *LWS mode-flag SP range specifier;
- mode-flag = "0" | "1";
- range specifier = npt range [SP byte-range] | byte-range;
- npt range = "npt" "=" npt time "-" npt time;
- npt time = <syntax defined in 0>;
- MT normative syntax for npt-time >;
- byte range = "bytes" "=" first byte pos "-" last byte pos;
- first byte pos = 1*DIGIT;
- last byte pos = 1*DIGIT.

It should be noted that literals, "npt" and "bytes", are case-sensitive.

Examples:

- `availableSeekRange.dlna.org: 0 bytes=214748364-224077003;`
- `availableSeekRange.dlna.org: 0 npt=00:05:30.12-00:10:34;`
- `availableSeekRange.dlna.org: 1 npt=00:05:30.12-00:10:34 bytes=214748364-224077003.`

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.20.9 If mode-flag is "0", then the following behaviours must be implemented by the HTTP Server Endpoint when responding with the `availableSeekRange.dlna.org` HTTP header:

- the HTTP server endpoint must implement rules in 7.5.4.2.16, 7.5.4.2.16.2 and 7.5.4.2.16.3 for Mode=0;
- the range-specifier (i.e., npt-range and/or bytes-range) must map to the $[r_0, r_N]$ data range, in such a way that, if the HTTP server endpoint receives a request that specifies a bytes-range or npt-range that is inclusively within the server's range-specifier, then the HTTP server endpoint must be able to serve the requested data bytes;
- if the HTTP server endpoints receives a GET request that omits both range and `TimeSeekRange.dlna.org` must respond from the "live position", as defined in 7.5.4.2.16.2.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline clarifies the Mode=0 rules described in 7.5.4.2.16, requirement 7.5.4.2.16.2 by explaining how specific syntax tokens refer to the UCDAM data boundaries.

Essentially the npt and byte positions in the range-specifier indicate the valid data range for use in requests that use range or `TimeSeekRange.dlna.org`. Similarly, a request omits these headers results with content data from the live position.

7.5.5.2.20.10 An HTTP client endpoint operating under the "limited random access data availability" model must specify a valid value for the first-byte-pos and/or the starting npt-time tokens in a request that specifies the range and/or TimeSeekRange.dlna.org HTTP header.

M	A	DMP DMR	M-DMP	MIU	[39]

As a corollary, a request that explicitly indicates "0" for npt or a byte position only works if the HTTP server endpoint indicates "0" is valid through the availableSeekRange.dlna.org.

7.5.5.2.20.11 If mode-flag is '1', then the following behaviours must be implemented by the HTTP server endpoint when responding with the availableSeekRange.dlna.org HTTP header:

- the HTTP server endpoint must implement rules in 7.5.4.2.16.4 for Mode=1;
- if present, the npt-range must map to $[r_0, r_N]$ such that if the HTTP server endpoint receives a request that specifies an npt-range that is inclusively within the server's range-specifier (as indicated in availableSeekRange.dlna.org), then the HTTP server endpoint must be able to serve the requested data bytes in a timely manner;
- if present, the bytes-range map to a subset of $[r_0, r_N]$ such that if an HTTP server endpoint receives a request that specifies a bytes-range that is inclusively within the server's range-specifier (as indicated in availableSeekRange.dlna.org), then the HTTP server endpoint must be able to serve the requested data bytes in a timely manner;
- the npt-pos and byte-pos of "0" must be valid and must point to the beginning of the content;
- servers that receive a GET request that omits both range and TimeSeekRange.dlna.org must respond with content data from the absolute beginning (i.e., s_0) of the content.

A timely response is defined in 7.5.4.2.16.4.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline clarifies the Mode=1 rules described in 7.5.4.2.16 and 7.5.4.2.16.4, by explaining how syntax tokens refer to the UCDAM data boundaries.

Essentially the byte range in the range-specifier indicate the data range for use in requests that use range where the response will return in a timely manner.

Unlike range, responses with TimeSeekRange.dlna.org are timely for the entire content binary.

Unlike Mode=0, a request that omits the range and TimeSeekRange.dlna.org headers results with content data from the beginning of the content binary, which is required to be static and non-changing.

7.5.5.2.20.12 If mode-flag is '1', then the HTTP server endpoint must be able to respond with the requested data bytes, even if the request specifies a bytes-range (in the HTTP request) that is not inclusively within the server's range-specifier (as indicated in the HTTP server's responses with availableSeekRange.dlna.org).

M	A	DMS +PU+	M-DMS	n/a	[39]

The content data mapped by range-specifier is where timely responses are mandatory.

The content data for $[s_0, s_N]$ is always available for transmission although the computational complexity to perform responses for the portions outside of the server's indicated range-

specifier (in the availableSeekRange.dlna.org header value) can cause in responses that are not timely.

7.5.5.2.20.13 If mode-flag is '1' and the HTTP server endpoint responds with data bytes that is not inclusively within the server's range-specifier (as indicated in the HTTP server's responses with availableSeekRange.dlna.org), then the HTTP server, may begin its response with content data after 27 s.

O	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.20.14 If an HTTP server endpoint supports the TimeSeekRange.dlna.org HTTP header for a content binary that operates under the "limited random access data availability" model, it must provide the npt-range in the availableSeekRange.dlna.org HTTP header field of the target response to an HTTP request with the getAvailableSeekRange.dlna.org header HTTP header field.

M	A	DMS +PU+	M-DMS	n/a	[39]

The availableSeekRange.dlna.org and getAvailableSeekRange.dlna.org HTTP headers apply in scenarios where the HTTP server and content are governed by the "limited random access data availability" model. Using these HTTP headers for other scenarios (such as those involving the "full random access data availability" model) is out-of-scope.

7.5.5.2.20.15 If an HTTP server endpoint supports the range HTTP header for a content binary that operates under the "limited random access data availability" model, it must provide the bytes-range in the availableSeekRange.dlna.org HTTP header field of the target response to an HTTP request with the getAvailableSeekRange.dlna.org header HTTP header field.

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.20.16 If the mode-flag=0, then an HTTP client endpoint must not assume any meaning for an npt value 0 or a byte position of 0 returned in an availableSeekRange.dlna.org HTTP header (i.e. if mode-flag=0, then npt-pos=0 and byte-pos=0 have undefined behaviours under most situations. An HTTP client endpoint that specifies a byte-pos=0 or npt-pos=0 must do so if the server's range-specifier includes 0 in the range.)

M	C	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

When the mode-flag is zero, then the "beginning of the content" becomes undefined. Clients are not to assume that npt or byte positions of 0 map to the beginning of the content. Furthermore, clients are not to assume that a GET request with a range or TimeSeekRange.dlna.org that specifies 0 will succeed.

7.5.5.2.21 MT HTTP mapping for byte-based seek and time-based seek

7.5.5.2.21.1 The range HTTP header must be used as the mechanism for byte-based seek operations on the HTTP transport protocol. The seek-able data range must be equivalent to the random access data range of $[r_0, r_N]$.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[39]

Byte-based seek is a generic, transport layer term that applies only to streaming transfers. The range header is a specific transport layer mechanism for HTTP.

It should be noted that the range header is valid for Interactive and background transfers, as described in 7.5.5.4.3 and 7.5.5.5.3. However, the byte-based seek operation is only valid for streaming transfers.

7.5.5.2.21.2 The TimeSeekRange.dlna.org HTTP header must be used as the mechanism for time-based seek operations on the HTTP transport protocol. The seekable data range must be equivalent to the random access data range of $[r_0, r_N]$.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[39]

Time-based seek is a generic, transport layer term that applies only to streaming transfers. The TimeSeekRange.dlna.org header is a specific transport layer mechanism for HTTP.

It should be noted that the time-based seek operation is valid only for streaming transfers. Using the TimeSeekRange.dlna.org header is expressly prohibited for background and interactive transfers.

7.5.5.2.22 MT HTTP header – Range (server)

7.5.5.2.22.1 An HTTP server endpoint must support the range header field defined in [39] for the HTTP GET and HEAD methods and follow the rules that are listed in the guidelines below.

M	L	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

HTTP server endpoints that receive an HTTP request with a range header field are expected to respond in a specific manner. The rules below describe what an HTTP server needs if the URI can or cannot support requests with a specified range.

7.5.5.2.22.2 If an HTTP server endpoint receives the range HTTP header in a HEAD request, then the HTTP server may respond without the Content-Range HTTP header.

O	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

HTTP server endpoints are required to support the HEAD method. When an HTTP server gets a HEAD request with the range option, the HTTP server may omit the Content-Range.

7.5.5.2.22.3 The notation of range header field for DLNA media transport is a subset of the allowed syntax for [39], as stated below:

- range-line = "Range" *LWS ":" *LWS range specifier;
- range specifier = byte range specifier;
- byte range specifier = bytes unit "=" byte range set;

- bytes unit = "bytes";
- byte range set = byte range spec;
- byte range spec = first byte pos "-" [last byte pos];
- first byte pos = 1*DIGIT;
- last byte pos = 1*DIGIT.

It should be noted that the literal, "bytes", is case-sensitive.

Examples:

- Range: bytes=1539686400-
- Range: bytes=1539686400-1540210688

M	L	DMS DMP DMR DMP+ +PU+ +UP+ +DN+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[39]

This guideline simplifies the implementation of HTTP server endpoints by requiring only a subset of the allowed Range syntax afforded by [39]. Essentially DLNA implementations of HTTP clients can only assume that a DLNA implementation of an HTTP server will support range values that indicate the first byte index and an optional last byte index. In summary, this restriction means that only a single range must be used within the range header field.

7.5.5.2.22.4 If an HTTP server endpoint returns data including the requested range (target response), it must specify the Content-Range header field. Furthermore, the HTTP response code must be: 206 (partial content).

Example of Content-Range header.

- Content-Range: bytes 1539686400-1540210688/9238118400

M	L	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

These guidelines obligate an HTTP server endpoint to respond with either a 206 response code (for a request that can be honoured) or a 416 response code (for requests that cannot be honoured). These guidelines simplify HTTP client endpoint implementations as they limit the HTTP server endpoint to more predictable behaviour.

Section 14.16 in [39] has examples on this guideline usage.

This guideline does not explain the relationship between first-byte-pos & last-byte-pos and the s_0 & s_N data boundaries. For example, if first-byte-pos=50 and last-byte-pos=200, then the response entity body will contain 151 bytes. The first byte of the entity-body will correspond to the 51st byte of the complete file and the last byte of the entity body will correspond to the 201st byte, relative to the complete content binary. This scenario does not at all imply that the 51st or 201st bytes (of the complete content binary) have any specific relationship to the s_0 or s_N data boundaries. Building on this example, if the s_N -increasing flag is true, then the complete binary is still increasing in size. This results in a scenario where the 201st byte does not map to the s_N data boundary.

7.5.5.2.22.5 If the HTTP server endpoint uses the Content-Range HTTP header, then the provided values must be accurate with respect to the entity response body. Specifically,

- the value indicating the first-byte-pos must properly match the first byte of the response entity body:

- the first-byte-pos in the response must match the first-byte-pos in the request message;
- the value indicating the last-byte-pos must properly match the last byte of the response entity body;
- the value indicating the instance-length must indicate the length of the entire content binary or asterisk (*) if unknown.

It should be noted that "accurate with respect to the entity response body" means that the guideline explains that the values used for the Content-Range header's first-byte-pos and last-byte-pos tokens need to correspond to the actual content data that is returned in the response entity body.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.22.6 If the requested range is not valid for the resource with a URI specified in the HTTP request, the HTTP server endpoint must respond with the HTTP response code of: 416 (requested range not satisfiable).

When encountering syntax errors with the range HTTP header, the HTTP server must use the HTTP response code 400 (bad request).

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.22.7 If a HTTP request with a range header for the URI can never be processed/satisfied by the HTTP server endpoint, for example, in the case of real-time transcoding or live content, the HTTP server endpoint must respond with 406 (not acceptable).

M	L	MS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

These guidelines clarify the HTTP range header support (which is mandatory in [39]). DLNA provides a limited number of exceptions from this requirement to meet the needs of live broadcast and transcoded content.

It should be noted that using a range header with a value of 'bytes=0-' is still using a range request.

7.5.5.2.22.8 If an HTTP server endpoint can support HTTP requests with a specified range for a particular URI, then the HTTP server endpoint should support persistent connections (HTTP/1.1) for that URI.

S	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

Content that is requested with the range option can often get many range requests in a short period of time, potentially causing content serving devices to run out of available sockets.

7.5.5.2.22.9 An HTTP server endpoint should accept and honour an HTTP/1.0 GET or HEAD requests for media transfers with the range header field.

S	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

Although the range option is not covered in the HTTP/1.0 specification, HTTP/1.0 clients can still benefit from having the ability to issue GET requests with a specified range.

7.5.5.2.22.10 An HTTP server endpoint should support HTTP requests that specify the range HTTP header by implementing behaviour that returns the requested entity-body with the 206 (partial content) HTTP response code.

S	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

HTTP server endpoints are allowed to respond with HTTP error code 406 (not acceptable) to indicate that specified HTTP headers (for example, range) are never valid for the specified URI. However, the recommended behaviour is to honour the HTTP request because many HTTP client endpoints rely on the range HTTP header to implement features like pause/resume, seek, and trick modes.

7.5.5.2.22.11 HTTP client endpoints must not issue an HTTP GET and HEAD requests with the range HTTP header for content binaries that do not explicitly state support for range requests via the 4th field op-Param as defined in guideline.

The syntax definition of op-param must be as follows.

M	L	DMP DMR +DN+ DMPr	M-DMP M-DMD	MIU	[39]

This prohibition includes HTTP GET and HEAD requests with range: bytes=0-.

7.5.5.2.23 HTTP range requirements for printing system usages

7.5.5.2.23.1 The HTTP server endpoint of a +PR1+ must implement the range header by sending responses that include the Content-Range header for requested images. Returning 406 (not acceptable) is expressly prohibited.

M	A	+PR1+	n/a	n/a	[39]

DMPr devices will be able to print more efficiently when the HTTP server fully supports the range header.

7.5.5.2.23.2 The HTTP client endpoint of a DMPr may use the range header to request portions of images when printing.

O	A	DMPr	n/a	n/a	[39]

7.5.5.2.23.3 The HTTP server endpoint of a printing control point must implement the range header by sending responses that include the Content-Range header for requested XHTML documents. Returning 406 (not acceptable) is expressly prohibited.

M	A	+PR1+ +PR2+	n/a	n/a	[39]

7.5.5.2.23.4 The HTTP client endpoint of a DMP_r must be able to print images without the use of the range header.

M	A	DMP _r	n/a	n/a	[39]

7.5.5.2.24 MT HTTP time-based seek (server)

7.5.5.2.24.1 HTTP server endpoints should support the TimeSeekRange.dlna.org HTTP header field, which is defined by DLNA, for the streaming transport of DLNA audio/visual and audio media.

S	A	DMS +PU+	M-DMS	n/a	[39]

HTTP requests with the range header field do not provide a very accurate experience when seeking to playback positions in variable bitrate encoded content. This HTTP header provides a way for DLNA HTTP clients to request an HTTP server to send the content bytes for a specified range of time.

It should be noted that HTTP clients may also use seek operations to implement a forward/backward variable speed playback capability.

7.5.5.2.24.2 If an HTTP server endpoint receives the TimeSeekRange.dlna.org HTTP header in a HEAD request for streamed transfer, then it may respond without the TimeSeekRange.dlna.org HTTP header.

O	C	DMS +PU+	M-DMS	n/a	[39]

All HTTP server endpoints are required to support the HEAD method. When a streaming HTTP server endpoint gets a HEAD request with this option, the HTTP server may omit it from the response.

7.5.5.2.24.3 The notation of the TimeSeekRange.dlna.org header field is defined as follows:

- TimeSeekRange-line = "TimeSeekRange.dlna.org" *LWS ":" *LWS range specifier;
- range specifier = npt range [SP bytes-range];
- npt range = "npt" "=" npt time "-" [npt time] [instance-duration];
- instance-duration = "/" (npt-time | "");
- npt time = <syntax as defined in 0>;
- MT normative syntax for npt-time>;
- bytes range = "bytes" "=" first byte pos "-" last byte pos instance-length;
- first byte pos = 1*DIGIT;
- last byte pos = 1*DIGIT;
- instance-length = "/" (1*DIGIT | "").

It should be noted that literals, "npt" and "bytes", are case-sensitive.

The npt-range specifies the range in normal playing time as found in 3.6 of [33] with the exception of the concept of the "now" literal. It is used in the request and response.

The bytes-range specifies the range in bytes and it is used only in the response. Refer to 7.5.5.2.24.5.

The instance-duration specifies the duration of an entire content binary and is mandatory in the response and prohibited in the request. The asterisk "*" character means that the instance-duration is unknown at the time when the response was generated. Refer to 7.5.5.2.24.5 for more information.

The instance-length specifies the byte length of an entire content binary and bytes-range must include instance-length in the response. The asterisk "*" character means that the instance-length is unknown at the time when the response was generated. Refer to 7.5.5.2.24.5 for more information.

Examples:

- TimeSeekRange.dlna.org: npt=335.11-336.08
- TimeSeekRange.dlna.org: npt=00:05:35.3-00:05: 37.5

Specifying the range value in the combination of npt-sec and npt-hhmmss, for example, 335.11-00:05:37.5, is allowed, but not recommended.

M	A	DMS DMP DMR +PU+ +DN+	M-DMS M-DMR M- DMD	MIU	[39] [33]

The field value specifies the requested range of a resource in absolute time. The range is specified by the start point and the endpoint. If the endpoint is omitted, it means the end of the resource is specified. This is similar to byte range specifier used for the range header field.

7.5.5.2.24.4 If an HTTP server endpoint returns data (target response) including the requested time range, it must return the content bytes for a time range that starts at or before the requested start point and ends at or after the requested endpoint. The following exceptions are allowed:

- HTTP server endpoint may ignore the requested endpoint and return the range data up to the end of the content data;
- HTTP server endpoint may round up or down to one decimal place, the npt time values specified in the HTTP response TimeSeekRange.dlna.org, compared to the actual returned data in the response body.

Examples:

- TimeSeekRange.dlna.org: npt=335.1-336.1/40445.4
- TimeSeekRange.dlna.org: npt=00:05:35.2-00:05:38.1/*

If the requesting endpoint specifies a time beyond the end of the content data, then the HTTP server endpoint must return the range data to the end of the content data.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline obligates an HTTP server to respond with a time range (of bytes) that is close to the time range specified in the request.

7.5.5.2.24.5 If an HTTP server endpoint supports both byte-based seek and time-based-seek operations for a resource, it must specify a byte-range value as well as an npt-range value in the HTTP Target Response to the HTTP request with the TimeSeekRange.dlna.org header field, unless the following exception applies.

Exception:

- if all of the following conditions are true, then the npt-range must be included (i.e. server is permitted to omit the bytes-range token in the TimeSeekRange.dlna.org, if the data range is unknown);
- the data access model is the "limited random access data availability" model;
- the mode-flag (as indicated in availableSeekRange.dlna.org) is "1";
- the request omitted the range header and only specified TimeSeekRange.dlna.org. (It should be noted that the range header always takes precedence over the TimeSeekRange.dlna.org, as indicated in 7.5.5.3.17.)

When specified, the npt-range and byte-range must include instance-duration and instance-length respectively.

Examples:

- TimeSeekRange.dlna.org:
npt=335.1-336.1/40445.4 bytes=1539686400-1540210688/304857907200
- TimeSeekRange.dlna.org:
npt=00:05:35.2-00:05:38.1/* bytes=1539686400-1540210688/*

M	A	DMS +PU+	M-DMS	n/a	[39]

The time-based-seek capability is useful for seeking to playback positions in variable bit rate encoded content; but it is not useful to retrieve subsequent data blocks. For trick mode playback after an initial time-based-seek, one should use multiple HTTP GET requests with the range header field to retrieve subsequent content data. To support this functionality, byte range value is also specified in addition to time range value for the response to a Time-Based-Seek only when byte-seek is also supported for the resource.

This functionality requires an HTTP server (that supports both TimeSeekRange.dlna.org and range) to specify both the time-range and a byte-range in the HTTP response's headers.

7.5.5.2.24.6 If an HTTP server endpoint returns data (target response) from a GET request with time range, then the response entity body data should start at a decoder friendly point (for example, the start of the GOP in a video sequence).

S	A	DMS +PU+	M-DMS	n/a	[39]

This guideline recommends that stream segments returned by servers should start, if possible, with a recognizable decoding entry point, such as the headers in a group of pictures (GOP).

7.5.5.2.24.7 If an HTTP server endpoint returns data (target response) from a GET request with time range, it must use the TimeSeekRange.dlna.org header field to indicate time range of the content data that is returned in the HTTP response. Furthermore, the HTTP response code must be 200 (OK).

The npt-range must include the instance-duration.

Examples:

- TimeSeekRange.dlna.org: npt=335.10-336.10/40445.4
- TimeSeekRange.dlna.org: npt=00:05:35.3-00:05: 37.5/*

M	A	DMS +PU+	M-DMS	n/a	[39]

These guidelines obligate an HTTP server to respond with either a 200 response code (for a time-range request that can be honoured) or a 416 response code (for time-range requests that cannot be honoured). The guideline simplifies HTTP client implementations as it places limits on the HTTP server to provide more predictable behaviour.

7.5.5.2.24.8 If an HTTP server endpoint supports the TimeSeekRange.dlna.org header and the requested time range is not valid for the resource with URI specified in the HTTP GET request, then the HTTP streaming server must respond with the HTTP response error code of: 416 (requested range not satisfiable).

Interpretation of not valid includes the following types of errors.

- The requested time range is not within the time boundaries of the actual content.

M	A	DMS +PU+	M-DMS	n/a	[39]

The HTTP server indicates support for the TimeSeekRange.dlna.org header in the DLNA.ORG_OP parameter which is in the contentFeatures.dlna.org in 7.4.1.3.18.

7.5.5.2.24.9 If an HTTP server endpoint supports the TimeSeekRange.dlna.org header and the requested time range is not syntactically correct, then the HTTP streaming server must return the HTTP response error code of 400 (bad request).

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.24.10 If an HTTP server endpoint supports the TimeSeekRange.dlna.org header and any HTTP request with TimeSeekRange.dlna.org for the specified URI can never be processed/satisfied by the HTTP server (for example, in the case of real-time transcoding or live contents), then the HTTP streaming server must return the HTTP response error code of 406 (not acceptable).

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.2.24.11 HTTP client endpoints must not issue an HTTP GET and HEAD requests with the TimeSeekRange.dlna.org HTTP header for content binaries that do not explicitly state support for TimeSeekRange.dlna.org request via the 4th field op-param as defined in 7.4.1.3.18.1.

M	L	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

7.5.5.2.25 MT HTTP chunked transfer coding

7.5.5.2.25.1 HTTP servers endpoints may use chunked transfer coding in response to HTTP/1.1 GET requests.

O	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

Chunked transfer coding is an HTTP response encoding methodology that can only be used in response to HTTP/1.1 requests by HTTP/1.1 servers.

7.5.5.2.25.2 HTTP server endpoints must not use chunked transfer coding in response to HTTP/1.0 GET requests.

M	R	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.25.3 HTTP client endpoints may use chunked transfer coding in HTTP/1.1 POST requests.

O	R	+UP+	M-DMU	MIU	[39]

7.5.5.2.26 MT baseline transport – HTTP client endpoints

7.5.5.2.26.1 HTTP client endpoints used for media transport purposes must implement HTTP/1.0, HTTP/1.1, or both.

M	L	DMP DMR DMPr +DN+	M-DMP M-DMD	MIU	[39]

HTTP client endpoints are restricted to HTTP versions that HTTP server endpoints are prepared to support.

7.5.5.2.26.2 HTTP client endpoints must not report a higher version of HTTP than is actually supported by the implementation.

M	R	DMP DMR DMPr +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

For example an HTTP client endpoint that does not support chunked transfer coding responses must never issue an HTTP/1.1 GET request.

7.5.5.2.26.3 HTTP/1.1 client endpoints must be able to process HTTP/1.1 chunked transfer coding responses.

M	R	DMP DMR DMPr +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

When making HTTP/1.1 requests, it is important that HTTP client endpoints properly handle responses encoded with chunked transfer coding.

7.5.5.2.26.4 HTTP/1.1 client endpoints must be prepared to properly handle an HTTP response code of 1xx from HTTP servers, even when not expected.

M	R	DMP DMR DMP _r +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

A 1xx can be generated by the server regardless of whether or not the client issued an HTTP request encoded with chunked transfer coding.

See 10.1 of [39] for more information.

7.5.5.2.27 MT HTTP header – Range (client)

HTTP client endpoints must not use multiple range specifiers nor use suffix byte range spec (as defined in [39]) in HTTP requests.

M	L	DMP DMR DMP _r +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

This guideline simplifies the implementation of HTTP servers by not requiring support of multiple ranges or suffix byte range spec.

7.5.5.2.28 MT HTTP persistent connection usage for clients

7.5.5.2.28.1 HTTP/1.1 client endpoints should use HTTP/1.1 persistent connections.

S	R	DMP DMR DMP _r +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

Implementing this guideline reduces the setup/teardown load on HTTP server endpoints. Furthermore, HTTP server endpoints will be able to reserve the allocated socket for the requesting client. Clients that do not use HTTP/1.1 persistent connections may encounter a scenario where an HTTP server endpoint does not answer the subsequent requests because it has run out of sockets.

7.5.5.2.28.2 HTTP/1.0 client endpoints must not use the CONNECTION: KEEP-ALIVE token for HTTP/1.0 transactions.

M	L	DMP DMR DMP _r +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[27]

See 7.5.5.2.17 for more information.

7.5.5.2.29 MT HTTP inactivity timeout

7.5.5.2.29.1 HTTP client endpoints should close persistent (HTTP/1.1) connections after completing all outstanding HTTP transactions and within 30 s of inactivity has passed.

S	A	DMP DMR DMP _r +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

This ensures that sockets do not remain consumed after a content transfer has successfully completed.

7.5.5.2.29.2 If an HTTP server detects 5 min of inactivity after a POST transaction, then the HTTP server should close persistent (HTTP/1.1) connections.

S	A	DMS	M-DMS	n/a	[39]

This would ensure that sockets do not remain consumed by an upload controller.

7.5.5.2.30 MT HTTP header parsing (client)

HTTP client endpoints must gracefully skip over unsupported HTTP header fields. Under no circumstances can an HTTP client fail to process a properly formatted HTTP response because of an unrecognized or unsupported HTTP header field.

M	R	DMP +DN+ DMR +UP+	M-DMP DMU	M-DMD M-	MIU	[39]

Incorrect HTTP header parsing has been the source of numerous compatibility issues during plugfest events.

7.5.5.2.31 MT HTTP maximum header size

HTTP client and server endpoints must use a total HTTP header size that is less than, or equal to, 8192 bytes (8 KB) when sending an HTTP request or HTTP response.

HTTP client and server endpoints must be tolerant of total HTTP header sizes that exceed 8192 bytes (8 KB) when receiving an HTTP request or HTTP response.

Tolerance of total HTTP header size means that the receiver can either "parse and interpret" or "parse and ignore" the HTTP headers that occupy a data range beyond the 8 KB boundary and up to the 128 KB boundary.

The total HTTP header size is the total number of bytes from the first byte in the start-line token and the last byte of the CRLF token, as used in the generic-message token defined in 4.1 of [39], as quoted in the syntax below.

- generic-message = start-line *(message-header CRLF) CRLF [message-body]

M	L	DMS DMP +UP+ +PR2+	DMR +PU+ +DN+ +PR1+	M-DMS DMD	M-DMP M-DMU	MIU	[39]

This provides a reasonable assumption as to how much memory is necessary for all HTTP headers in an HTTP request or response.

This guideline also ensures tolerance of future HTTP client and server endpoints, where the HTTP header sizes may exceed 8 KB.

If determined to be necessary, future DLNA guidelines can define a parameter in the USER-AGENT to indicate a DLNA version for governing total HTTP header sizes.

7.5.5.2.32 MT HTTP status code precedence

If a DLNA guideline specifies an HTTP status code that involves a DLNA defined HTTP header, then the DLNA guideline's HTTP status code must take precedence over those specified in [39].

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

As a general rule, HTTP status codes for DLNA defined HTTP headers take precedence over HTTP status codes. For standard HTTP headers, [39] status codes apply. For HTTP responses that involve DLNA defined HTTP headers, the DLNA specified status code applies when there exists an ambiguity between using a DLNA-specified error code and an error code made by [39]. Whenever possible, DLNA guidelines align with [39].

7.5.5.2.33 MT transfer mode indication

7.5.5.2.33.1 The syntax for the transferMode.dlna.org HTTP header value is as follows:

- transferMode-line= "transferMode.dlna.org" *LWS ":" *LWS mode;
- mode = "Streaming" | "Interactive" | "Background".

A value of "background" for an HTTP request indicates that the HTTP client would like the transfer of content to be a background transfer.

A value of "interactive" for an HTTP request indicates that the HTTP client is requesting an interactive transfer.

A value of "streaming" for an HTTP request indicates that the HTTP client wishes a streaming transfer of the content.

A value of "background" for an HTTP response indicates that the HTTP server is attempting a background transfer of the content.

A value of "interactive" for an HTTP response indicates that the HTTP server is attempting an interactive transfer of the content.

A value of "streaming" for an HTTP response indicates that the HTTP server is attempting a streaming transfer of the content.

It should be noted that the mode token values ("streaming", "interactive", and "background") are case-sensitive values.

Example

- transferMode.dlna.org: Interactive

M	A	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M-DMD M-DMU	MIU	[39]

This header carries the transfer mode as listed in the introduction.

7.5.5.2.33.2 An HTTP server that receives an HTTP request from an HTTP client of a DLNA endpoint that omits the transferMode.dlna.org header must treat it as equivalent to the header-value pair of transferMode.dlna.org:Streaming for content binaries of the AV or audio media classes or transferMode.dlna.org:Interactive for all other content binaries.

M	A	DMS	M-DMS	n/a	[39]

DLNA endpoint transfers that do not include this header should be treated as streaming transfers

7.5.5.2.33.3 The transferMode.dlna.org header must be used by HTTP clients and servers to convey the transfer mode information as specified in guideline 7.5.4.2.3.

M	C	DMS DMP DMR DMP+ +PU+ +DN+ +PR1+ +PR2+ +UP+	M-DMS M-DMP M- DMD M-DMU	MIU	[39]

This guideline clarifies the use of the transferMode.dlna.org, according to 7.5.4.2.3.

7.5.5.2.33.4 If an HTTP client requests a transfer mode that is not valid for the type of data that is being exchanged or is not supported by the server for the given content binary, the HTTP server must respond with error code 406 (not acceptable).

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

The list of transfer types for the given media is given in 7.5.4.2.3.2. However, a server may restrict the list further if the content warrants it. For example, for live captured content, background transfer may not be available. For details on the media transfer modes that are available for a particular content binary within a DMS structure, see 7.4.1.3.23.2.

7.5.5.2.33.5 All DLNA endpoints communicating with a v1.0 DMS must tolerate HTTP responses that omit the transferMode.dlna.org HTTP header.

Tolerate means that the HTTP client must "parse and ignore" or "parse and interpret" the HTTP response (i.e., all DLNA endpoints communicating with a v1.0 DMS must assume that the media will be returned without the transferMode.dlna.org HTTP header and using a best effort QoS level.)

M	C	DMP DMR DMP+ +DN+ +UP+	M-DMP M-DMD M- DMU	MIU	[39]

All v1.0 DMS media transfers were assumed to be for immediate rendering and at a default best effort QoS level. A v1.5 entity communicating with a v1.0 DMS must allow any media transfer to have these parameters.

7.5.5.2.34 MT caching directives for HTTP 1.0

7.5.5.2.34.1 If applicable-http-endpoints want to modify the default caching behaviour of intermediate HTTP caches, they must do so in a manner compliant with [27].

Applicable-http-endpoints specifically refers to HTTP server and client endpoints participating in transfer operations of cacheable content using

- HTTP/1.0; and
- GET requests and responses.

This definition of applicable-http-endpoints is valid only for this guideline.

M	C	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M- DMD M-DMU	MIU	[27]

Per HTTP/1.0 specifications, intermediate caching operations are allowed by default when content is transferred using the GET method.

7.5.5.2.34.2 If an HTTP server endpoint transfers non-cacheable content using HTTP/1.0 GET responses, then the HTTP server endpoint must prevent intermediate caching by including this directive amongst the HTTP headers in the response.

- Pragma: no-cache

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[27]

Non-cacheable content includes the following:

- live content and other forms of broadcast streams;
- content data that includes the TimeSeekRange.dlna.org or PlaySpeed.dlna.org HTTP headers;
- content binaries that are dynamically generated in such a way that the content binary streams can differ on two different transactions (for example, smart transcoding engines that perform transrating based on network throughput).

7.5.5.2.34.3 If HTTP server endpoints transfer audio or AV content binaries and the response includes one or both of listed HTTP headers, then the responses must prevent caching, according to 7.5.5.2.34.1 and 7.5.5.2.34.2.

- TimeSeekRange.dlna.org
- PlaySpeed.dlna.org

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	n/a

In HTTP 1.0 no method exists to signal the different bit stream variations that result from the use of time seek and play speed operations. For this reason, intermediate caching operations have to be avoided.

7.5.5.2.35 MT caching directives for HTTP 1.1

7.5.5.2.35.1 If applicable-http-endpoints want to modify the default caching behaviour of intermediate HTTP caches, they must do so in a manner compliant with [39].

Applicable-http-endpoints specifically refers to HTTP server and client endpoints participating in transfer operations of cacheable content using the following:

- HTTP/1.1; and
- GET requests and responses.

This definition of applicable-http-endpoints is valid only for this guideline.

M	C	DMS DMP DMR DMP+ +PU+ +DN+ +UP+ +PR1+ +PR2+	M-DMS M-DMP M-DMD M-DMU	MIU	[39]

Per HTTP/1.1 specifications, intermediate caching operations are allowed by default when content is transferred using the GET method.

7.5.5.2.35.2 HTTP server endpoints that transfer non-cacheable content using HTTP/1.1 GET responses must prevent intermediate caching by including among the HTTP response headers both of the following directives:

- Pragma: no-cache;
- Cache-control: no-cache.

M	C	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.2.35.3 If an HTTP server endpoint transfers audio or AV content binaries using HTTP/1.1 GET responses that include one or both of these HTTP headers, then such transfers should be marked as cacheable.

- TimeSeekRange.dlna.org
- PlaySpeed.dlna.org

S	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	n/a

Unlike HTTP/1.0, the newer version HTTP/1.1 includes caching directives that, when used, enable transparent caching even when the transferred objects include headers that modify the object's binary representation. In DLNA, time seek and play speed headers modify the binary representation of the object that is transferred using the HTTP protocol. The "vary" header described below can be used to restore transparent caching for these objects.

7.5.5.2.35.4 If an HTTP server endpoint transfers an audio or AV content binaries that permit variable play speed and time-based seek operations for cacheable content transported in an HTTP/1.1 GET response, then the HTTP server endpoint must include a "vary" HTTP header as defined in [39].

The "vary" header must list either or both of the following two arguments to inform caches of the corresponding supported operations.

- TimeSeekRange.dlna.org;
- PlaySpeed.dlna.org.

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

The vary header serves to indicate potential intermediate caches that certain headers create variants of the object binaries defined by its URI. For example, a 30-min media stream requested in full is a different object variant when only the first 3 min of the stream are requested using time seek operations.

NOTE According to [39] the vary header is included whenever the server responds to a client request for the CDS object. The vary header is included even if the request does not include time seek or play speed headers.

7.5.5.2.35.5 If applicable-http-endpoints want to supersede the default caching prohibition for POST transfers, they must do so in a manner compliant with [39].

Applicable-http-endpoints specifically refers to HTTP server and client endpoints participating in upload transaction using

- HTTP/1.1; and
- POST requests and responses.

M	C	DMS +UP+	M-DMS M-DMU	MIU	[39]

According to HTTP/1.1 specifications, intermediate caching operations are prohibited by default when content is transferred using the POST method.

7.5.5.2.36 MT/BCM HTTP header – peerManager.dlna.org

7.5.5.2.36.1 HTTP client endpoints may use the peerManager.dlna.org HTTP header to communicate the identity of the UPnP AV ConnectionManager service to the content source endpoint in HTTP GET requests.

O	A	DMR	n/a	n/a	[39]

The peerManager.dlna.org HTTP Header is not required. However, this feature is useful for HTTP clients that want to facilitate BCM on DMS with BCM support.

7.5.5.2.36.2 The value of the peerManager.dlna.org HTTP header must be the same value as the PeerConnectionManager, as described in UPnP AV architecture.

The notation of the peerManager.dlna.org HTTP header field is defined as follows:

- peerManager-line = "peerManager.dlna.org" *LWS ":" *LWS peer-connection-manager;
- peer-connection-manager = udn-token "/" serviceId-token;
- udn-token = <case-insensitive UDN of the UPnP AV MediaRenderer device>;
- serviceId-token = <case-sensitive ServiceID of the ConnectionManager service, associated with the MediaRenderer device identified by udn-token>.

Example

- peerManager.dlna.org: uuid:12345678123456781234567812345678/urn:schemas-upnp-org:serviceId:ConnectionManager

M	A	DMR	n/a	n/a	[39]

7.5.5.2.37 MT/BCM HTTP header – friendlyName.dlna.org

7.5.5.2.37.1 HTTP client endpoints may use the friendlyName.dlna.org HTTP header to communicate a user friendly name to the content source endpoint in HTTP GET requests.

O	A	DMP +UP+ +DN+	n/a	MIU	[39]

The friendlyName.dlna.org HTTP header is not required. However, this feature is useful for HTTP clients that want to facilitate BCM on DMS with BCM support.

7.5.5.2.37.2 The notation of the friendlyName.dlna.org header field is defined as follows:

- friendlyName-line = "friendlyName.dlna.org" *LWS ":" *LWS "friendly-name-token
- friendlyName-token = <case sensitive string, limited to 128 bytes in its UTF-8 encoded form>

M	A	DMP +UP+ +DN+	n/a	MIU	[39]

7.5.5.2.38 MT/BCM scid.dlna.org HTTP header

7.5.5.2.38.1 If a UPnP AV MediaServer supports BCM, then it must use the scid.dlna.org HTTP header in HTTP responses to identify the ConnectionID associated with the underlying transport layer connection.

M	A	DMS	M-DMS	n/a	[39]

7.5.5.2.38.2 The syntax of the scid.dlna.org HTTP header must be as follows:

- scidheader = "scid.dlna.org" *LWS ":" *LWS connection-id;
- connection-id = <the Connection ID value associated with the underlying transport connection>

M	A	DMS DMP DMR +UP+ +DN+	M-DMS	MIU	[39]

7.5.5.2.38.3 HTTP client endpoints may use scid.dlna.org in subsequent HTTP requests to identify a known and associated UPnP AV connection only if the HTTP requests are for the same URI.

O	A	DMP DMR +UP+ +DN+	M-DMP	MIU	[39]

This guideline allows content receivers to provide the connection ID in the transport layer requests. This guideline assumes the HTTP client of the content receiver acquired the connection ID in an earlier HTTP response. This mechanism may be used to combine logically related but separated transport layer requests for the same content URI.

7.5.5.3 HTTP transport – Streaming transfer guidelines, HTTP media transport for streaming transfer guidelines

7.5.5.3.1 MT streaming transferMode.dlna.org HTTP header

7.5.5.3.1.1 An HTTP client endpoint requesting a streaming transfer of data must specify the transferMode.dlna.org HTTP header and it must have a value of "streaming".

M	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

Version 1.0 clients will still request streaming transfers without using the transferMode.dlna.org HTTP header. Subclause 7.5.6.2.8.2 requires that servers that receive such a request treat it as equivalent to a request for a streaming transfer for audio and AV media classes

7.5.5.3.1.2 Unless a streaming HTTP server endpoint responds with an error response code, it must respond to HTTP HEAD or GET requests for a streaming transfer (as defined in 7.5.5.3.1.1) by sending a response that contains the transferMode.dlna.org HTTP header and gives it a value of "streaming".

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.3.2 MT HTTP play media operation

A streaming HTTP client endpoint must use the HTTP GET method when using the HTTP transport protocol for initiating the play media operation.

M	L	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

This guideline specifies the normative way to request content.

7.5.5.3.3 MT HTTP stop media operation

A streaming HTTP client endpoint should implement the stop media operation by disconnecting the TCP connection of the HTTP transaction.

S	L	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

This guideline recommends the way to stop a media stream. Although HTTP clients can technically stall the TCP connection, that technique is not recommended. The recommended technique makes better use of an HTTP server endpoint's platform resources, which may be limited.

It should be noted that HTTP client endpoints are required to visually and/or aurally stop rendering of content, although the continuation of data streaming from the HTTP server endpoint is permissible for data buffering/caching purposes.

7.5.5.3.4 MT HTTP pause/pause-release media operation

7.5.5.3.4.1 A streaming HTTP client endpoint should implement the pause media operation by one of the following methods.

- Disconnecting and seeking. Disconnecting the HTTP connection with the intent to use byte-base seek or time-based seek transport layer features as the mechanism for pause-release.
- Connection stalling. Suspending the reading of data from the HTTP connection. Resuming the reading of data from the HTTP connection is the mechanism for pause-release.

S	L	DMP DMR	M-DMP	MIU	[39]

This guideline recommends two ways to pause a media stream.

7.5.5.3.4.2 If a streaming HTTP client endpoint wants to pause the current media stream, it must first ensure that all of the necessary media operations and information are available to resume the play (pause-release) of the media stream.

M	A	DMP DMR	M-DMP	MIU	[39]

The ability to do a pause-release media operation depends on both the content receiver and the content source sharing support for at least one of the following HTTP transport features: byte-based seek, time-based seek, or connection stalling.

7.5.5.3.5 MT HTTP pause/pause-release media operation – Disconnecting and seeking method

7.5.5.3.5.1 If a streaming HTTP client endpoint performs a pause media operation using the disconnecting and seeking method, it must support the pause-release operation by using byte-based seek or time-based seek transport layer features. Before using this form of pause/pause-release mechanism, the streaming HTTP client endpoint must verify that the content source supports the necessary transport layer feature.

M	A	DMP DMR	M-DMP	MIU	[39]

The HTTP connection may be closed for different reasons. If the connection is disconnected, the streaming HTTP client endpoint may use a seek media operation (see 7.5.5.3.7) to pause-release the playback.

7.5.5.3.5.2 If a streaming HTTP client endpoint supports the pause media operation using the disconnecting and seeking method, it should support the byte-based seek transport layer feature.

S	A	DMP DMR	M-DMP	MIU	[39]

It is recommended that a content source supports byte-based seek, as defined in 7.5.5.2.22.

7.5.5.3.5.3 If a streaming HTTP client endpoint supports the pause media operation using the disconnecting and seeking method, it may perform the pause media operation by first suspending the reading of data from the HTTP connection (as described for the connection stall method). When the streaming HTTP client endpoint detects a TCP-layer disconnect, it may perform the pause-release media operation using the time-based seek or byte-based seek transport layer feature that is supported by the content source.

O	A	DMP DMR	M-DMP	MIU	[39]

Using TCP flow control to stall/pause the flow of data can enable quick pause-release behaviour. Content sources are recommended to support the connection stalling method, in addition to byte-based seek or time-based seek transport layer features.

7.5.5.3.6 MT HTTP pause/pause-release media operation – Connection stalling method

7.5.5.3.6.1 If a streaming HTTP client endpoint performs a pause media operation using the connection stalling method it must verify the http-stalling parameter in the 4th field of the res@protocolInfo is present and set to true for a content binary.

M	A	DMP DMR	M-DMP	MIU	[39]

A pause media operation initiated by the connection stalling method does not require the use of time-based seek or byte-based seek since the content source maintains the TCP connection, using standard TCP flow control. Content sources may choose to support only the connection stalling method for some content binaries, such as those created by dynamic, real-time transcoding.

7.5.5.3.6.2 When the HTTP connection is lost for any reason, the streaming HTTP client endpoint may attempt to use time-based seek or byte-based seek transport layer features to pause-release the media stream.

O	A	DMP DMR	M-DMP	MIU	[39]

This guideline assumes both the content receiver and content source support the same transport layer features.

7.5.5.3.6.3 If the http-stalling flag is true for a content binary, then the streaming HTTP server endpoint must allow connection stalling for an indefinite amount of time on that content binary.

Equivalently, streaming HTTP server endpoints that support the connection stalling method for a content binary must maintain the HTTP connection and must not use an HTTP connection inactivity timeout to terminate the HTTP connection.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline prohibits using an HTTP-inactivity timeout to terminate HTTP connections that are being paused through connection stalling.

The guideline permits the streaming HTTP server endpoint to terminate HTTP connections for the following scenarios:

- when the content receiver terminates the HTTP connection;
- the underlying TCP transport session is broken or disconnected;
- system events on the streaming HTTP server endpoint: user-initiated termination of streams, scheduled recording events, configurable policies for idle connections, etc.

Although this guideline requires a streaming HTTP server endpoint to allow connection stalling for an indefinite period of time, a streaming HTTP server endpoint can provide users with the ability to terminate the HTTP connections. Many details in this area are out of scope, but this guideline accounts for the following types of possibilities.

- A local UI, associated with the content source, allows the user to manually terminate the HTTP connections.
- The content source has user-configurable policies that can override the default behaviour of indefinite connection stalling by terminating HTTP connections that have been inactive for a lengthy time. These guidelines do not define a minimum time but the suggested minimum HTTP inactivity timeout is 5 min.
- UPnP AV MediaServer control points invoke CMS:ConnectionComplete to terminate connections.

7.5.5.3.7 MT HTTP seek media operation

A streaming HTTP client endpoint should implement the seek media operation by using the range or TimeSeekRange.dlna.org header fields, if those header fields are supported by the HTTP server for the content binary.

S	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

This guideline recommends two ways to implement the seek operation on content. The method used by the client is conditional on whether or not the HTTP server supports the capability for that content.

See 7.4.1.3.18 and 7.4.1.3.28 for information on discovering server seek capabilities.

See 7.5.5.2.22 and 7.5.5.2.24 for more information on range and TimeSeekRange.dlna.org header fields.

7.5.5.3.8 MT HTTP fast forward scan media operation

If a streaming HTTP client endpoint wants to perform a fast forward scan media operation (positive play speed greater than 1x), then it should use one of these methods, provided that the given header fields are supported by the server:

- issuing multiple HTTP GET requests with a specified range header field;
- issuing multiple HTTP GET requests with a specified TimeSeekRange.dlna.org header field;
- issuing a single HTTP GET request with a specified PlaySpeed.dlna.org header field.

S	A	DMP DMR	M-DMP	MIU	[39]

Forward scan and backward scan operations fall into a category of playback capabilities referred to as trick modes. These guidelines specify three general techniques for implementing trick modes.

The first technique is to issue multiple HTTP requests with specified byte ranges, such that the byte data can be rendered sequentially, giving the effect of a trick mode playback. With this technique, the HTTP client endpoint is responsible for specifying the appropriate byte ranges that will achieve the desired effect.

The second technique is a variant of the first, and it works by requesting the HTTP server to return time ranges (instead of byte ranges). In this technique, the HTTP client endpoint is responsible for choosing the appropriate time ranges that achieve the desired effect.

The third technique works by having the HTTP client endpoint instruct the HTTP server endpoint to return byte data that is already time scaled for a particular play speed.

See 7.5.5.2.22, 7.5.5.2.24, and 7.5.5.3.16 for more information on range, TimeSeekRange.dlna.org, and PlaySpeed.dlna.org header fields.

7.5.5.3.9 MT HTTP streaming slow forward scan media operation

If a streaming HTTP client endpoint wants to perform a slow forward scan media operation (positive play speed less than 1x), then it should use one of these methods:

- issuing multiple HTTP GET requests with a specified range header field; or
- issuing a single HTTP GET request with a specified PlaySpeed.dlna.org header field.

S	A	DMP DMR	M-DMP	MIU	[39]

7.5.5.3.10 MT HTTP streaming fast backward scan media operation

If a streaming HTTP client endpoint wants to perform a fast backward scan operation (negative play speed less than -1x), then it should use one of these methods:

- issuing multiple HTTP GET requests with a specified range header field;
- issuing multiple HTTP GET requests with a specified TimeSeekRange.dlna.org header field;
- issuing a single HTTP GET request with a specified PlaySpeed.dlna.org header field.

S	A	DMP DMR	M-DMP	MIU	[39]

7.5.5.3.11 MT HTTP streaming slow backward scan media operation

If a streaming HTTP client endpoint wants to perform a slow backward scan media operation (negative play speed greater than or equal to -1x), then it should use one of these methods:

- issuing multiple HTTP GET requests with a specified range header field; or
- issuing a single HTTP GET request with a specified PlaySpeed.dlna.org header field.

S	A	DMP DMR	M-DMP	MIU	[39]

7.5.5.3.12 MT HTTP streaming scan media operations

7.5.5.3.12.1 If a streaming HTTP client endpoint wants to stop a normal playback stream in order to start a scan operation playback using the range header under conditions where 7.5.5.2.4.6 applies, then it should close the original HTTP connection before issuing a GET request with the range header to perform scan operations (a.k.a. trick modes). After closing the original connection, the streaming HTTP client endpoint should open a new HTTP connection for the scan operation. Please observe that the new HTTP connection may be a persistent connection, which allows the client to issue multiple GET requests on a single HTTP connection.

S	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

It should be noted that transitions from scan operations to normal playback may be achieved by making open ended (i.e. last-byte-pos value in range header is absent) GET requests with the range option.

7.5.5.3.12.2 If a streaming HTTP client endpoint wants to stop a normal playback stream in order to start a scan operation playback using the PlaySpeed.dlna.org header under conditions where 7.5.5.2.4.6 applies, then it should close the original HTTP connection before issuing a GET request with the PlaySpeed.dlna.org header to perform scan operations (a.k.a. trick modes). After closing the original connection, the streaming HTTP client should open a new HTTP connection for the scan operation. It should be observed that the new HTTP connection may be a persistent connection, which allows the client to issue multiple GET requests on a single HTTP connection

S	A	DMP DMR	M-DMP	MIU	[39]

It should be noted that transitions from scan operations to normal playback may be achieved by closing the HTTP connection that provides scan operations and opening a new HTTP connection for normal playback speed.

7.5.5.3.13 MT HTTP streaming download media operation

An HTTP client endpoint must initiate a streaming download media operation with the HTTP GET method when using the HTTP transport protocol for content and it must specify the transferMode.dlna.org HTTP header with a value of "streaming".

M	L	+DN+	M-DMD	MIU	[39]

7.5.5.3.14 MT HTTP prohibited operations for streaming download

7.5.5.3.14.1 An HTTP client endpoint performing a streaming download media operation must not use any of the following media operations as specified in these guidelines.

Table 26 – TTP prohibited operations references

Operation	Reference
Pause	7.5.5.3.4 MT HTTP pause/pause-release media
Fast forward scan	7.5.5.3.8 MT HTTP fast forward scan media
Slow forward scan	7.5.5.3.9 MT HTTP streaming slow forward scan media
Fast backward scan	7.5.5.3.10 MT HTTP streaming fast backward scan media operation
Slow backward scan	7.5.5.3.11 MT HTTP streaming slow backward scan media operation

7.5.5.3.14.2

M	L	+DN+	M-DMD	MIU	[39]

7.5.5.3.15 MT HTTP media operation support within a profile

7.5.5.3.15.1 If the following conditions are true, then the HTTP client endpoint must support that media operation on all content with that media format profile:

- the HTTP client endpoint supports a media operation on a DLNA media format profile;
- the HTTP server endpoint supports the necessary protocol elements to support the media operation.

M	A	DMP DMR +DN+	M-DMP M-DMD	MIU	n/a

Necessary protocol elements may include optional parameters such as byte range and TimeSeekRange.dlna.org.

7.5.5.3.15.2 If an HTTP server endpoint supports byte range and/or TimeSeekRange.dlna.org capabilities on the content for a particular DLNA media format profile, it should support that on all content with that media format profile.

S	A	DMS +PU+	M-DMS	n/a	n/a

In some cases, such as transcoded or live content, it may be difficult to support byte range and/or TimeSeekRange.dlna.org.

7.5.5.3.16 MT HTTP PlaySpeed.dlna.org header

7.5.5.3.16.1 The streaming HTTP server endpoint may support the PlaySpeed.dlna.org HTTP header field, for the transport of AV and audio media class for variable speed streaming playback.

O	A	DMS +PU+	M-DMS	n/a	[39]

This allows HTTP clients to request the HTTP server to return content in a time-scaled form.

For example, a DLNA HTTP client can request a DLNA HTTP server to return DLNA content in a 4x playback speed. The HTTP server's response would send content that gives the appearance of 4x playback speed.

It should be noted that this HTTP header may be used by DLNA servers for content conforming to DLNA media format profiles and content that does not conform to DLNA media format profiles.

7.5.5.3.16.2 If a streaming HTTP server endpoint receives the PlaySpeed.dlna.org HTTP header in a HEAD request, then the HTTP server may respond without the PlaySpeed.dlna.org HTTP header.

O	A	DMS +PU+	M-DMS	n/a	[39]

HTTP servers are required to support the HEAD method. When an HTTP server gets a HEAD request with this option, the HTTP server may omit it from the response.

If HTTP clients need to determine if the PlaySpeed.dlna.org is supported for a URI, then the clients are supposed to use the mandatory getContentFeatures.dlna.org HTTP header and examine the DLNA.ORG_PS parameter.

The reason why HEAD responses do not require the use of PlaySpeed.dlna.org is that the computational effort to respond with only the headers for a request that includes PlaySpeed.dlna.org is significant.

7.5.5.3.16.3 If the streaming HTTP Endpoint uses the PlaySpeed.dlna.org HTTP header, then the Endpoint must use the following syntax for the HTTP header and its value:

- PlaySpeed-line = "PlaySpeed.dlna.org" *LWS " " *LWS play speed specifier;
- play speed specifier = "speed" "=" transport-play-speed;
- transport-play-speed = <use the same notation for the AVT.TransportPlaySpeed state variable defined in the UPnP AV Transport service type>.

It should be noted that the "speed" token is case-sensitive.

Examples:

- PlaySpeed.dlna.org: speed=10
- PlaySpeed.dlna.org: speed=-1/2

When encountering syntax errors with the PlaySpeed.dlna.org HTTP header, the HTTP server must use the HTTP response code 400 (bad request).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[39]

The field value specifies a play speed to scale content data of a resource. The value is represented as same as TransportPlaySpeed state variable defined by AV transport service type (for example, 5, 10, -1/2, -10, -3/2, etc.).

7.5.5.3.16.4 If the streaming HTTP server endpoint returns data (target response) for scaled content to be decoded for a scan operation (a.k.a. variable play speed), the HTTP response message must use the PlaySpeed.dlna.org HTTP header field to indicate the play speed of the scaled content. Furthermore, the HTTP response code must be 200 (OK).

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline requires the HTTP server to indicate if content bytes in the HTTP response represent content that has been time-scaled.

7.5.5.3.16.5 If any requested PlaySpeed.dlna.org for the specified URI can never be processed/satisfied by the streaming HTTP server endpoint, for example, in the case of real-time transcoding or live contents, the streaming HTTP server endpoint must respond with 406 (not acceptable).

If an HTTP server supports the PlaySpeed.dlna.org header and the requested play speed is not valid for the resource with URI specified in the HTTP GET request, then the HTTP server must respond with the HTTP response error code 406 (not acceptable). Interpretation of not valid includes the following types of errors.

- The requested TransportPlaySpeed is not supported for the content.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline specifies the error code to be used in scenarios where the HTTP server cannot accommodate a request to time scale content.

The HTTP server indicates support for the PlaySpeed.dlna.org header in the DLNA_ORG_PS parameter which is in the contentFeatures.dlna.org in 7.4.1.3.21.

7.5.5.3.16.6 The scaled data (returned by the streaming HTTP server endpoint as a target response) must be compliant to the media format profile indicated in the corresponding <res> element, obtained from a UPnP AV ContentDirectory service implementation.

M	A	DMS +PU+	M-DMS	n/a	[39] [55]

This guideline obligates an HTTP server endpoint to return content bytes that are conformant to the characteristics described for that content by the associated UPnP AV ContentDirectory service. For example, if the content is exposed as MPEG2_PS_NTSC content, the returned content must conform to the MPEG syntax defined for that media format profile.

7.5.5.3.17 MT combined range, time based seek, and play speed HTTP requests

7.5.5.3.17.1 If a streaming HTTP server endpoint receives an HTTP GET request with both the PlaySpeed.dlna.org and the TimeSeekRange.dlna.org header fields, the endpoint must understand that a scan operation is requested for the specified time range.

If the streaming HTTP server endpoint can never process either or both time-scaling and time-seek, the error code for time-scaling, 406 (not acceptable) must be returned.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline covers what a streaming HTTP server endpoint needs to do if it receives an HTTP request for both time-based seek and play speed. This guideline also infers how a streaming HTTP client endpoint can expect an HTTP server endpoint to behave.

7.5.5.3.17.2 If a streaming HTTP server endpoint receives an HTTP GET request with Range and (TimeSeekRange.dlna.org or PlaySpeed.dlna.org) header fields, then the range header field must take the highest precedence and the server must ignore the other time seek and play speed fields.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline covers what a streaming HTTP server endpoint needs to do if it receives an HTTP request with range and other DLNA fields for play speed or time-based seek. This guideline also infers how a streaming HTTP client endpoint can expect an HTTP server endpoint to behave.

7.5.5.3.18 MT HTTP header: realTimeInfo.dlna.org

7.5.5.3.18.1 HTTP client endpoints may use realTimeInfo.dlna.org in GET or HEAD requests to specify the desired policy for content delivery.

O	A	DMP DMR +DN+	M-DMP M-DMD	MIU	[39]

By using this header, an HTTP Client Endpoint (for example, DMP) informs an HTTP server endpoint (for example, DMS) the way it wants to receive content byte data (unmodified or in a real-time manner). In the second case, the loss of content quality (i.e., drop data bytes) is possible.

In the case of real-time delivery, it provides a hint to an HTTP client endpoint on how long it should pre-buffer content data bytes before beginning the playback.

7.5.5.3.18.2 If an HTTP client endpoint request contains the realTimeInfo.dlna.org header, then the HTTP server endpoint reply should include the realTimeInfo.dlna.org header.

S	A	DMS +PU+	M-DMS	n/a	[39]

An HTTP client endpoint (for example, DMP) must be ready to receive an undesired value for the max-delay-time from an HTTP server endpoint (for example, DMS).

It is worth mentioning that, for a live content, the max-lag-time value cannot be set as infinite.

7.5.5.3.18.3 If an HTTP server endpoint responds with a realTimeInfo.dlna.org header with an infinite max-lag-time (i.e., with value of "***"), then it must not drop and/or modify any portion of content byte data.

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.3.18.4 If an HTTP server endpoint responds with a realTimeInfo.dlna.org header with a finite max-lag-time, then it must not send stale data when the delay time is more than the max-lag-time. In this case an HTTP server endpoint must omit the Content-Length header from HTTP response.

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.3.18.5 An HTTP server endpoint should include the realTimeInfo.dlna.org header in each HTTP reply.

S	A	DMS +PU+	M-DMS	n/a	[39]

It is desirable to provide an HTTP client endpoint (for example, DMP) the information for the delivery manner each time.

7.5.5.3.18.6 If an HTTP server endpoint receives an HTTP GET or HEAD request with a range and `realTimeInfo.dlna.org` header, then an HTTP server endpoint must never reply with a finite max-lag-time parameter value.

M	A	DMS +PU+	M-DMS	n/a	[39]

This guideline obliges an HTTP server endpoint to not change the content data bytes in a reply to a request with a range header.

7.5.5.3.18.7 The notation for the `realTimeInfo.dlna.org` HTTP header is defined as follows:

- `realTimeInfo-line` = "realTimeInfo.dlna.org" *LWS ":" *LWS max-lag-time;
- max-lag-time= "DLNA.ORG_TLAG" "=" duration;
- duration = npt-sec | "**";
- ntp-sec= 1*DIGIT["."1*3DIGIT].

The literal, "DLNA.ORG_TLAG" is case-sensitive.

The max-lag-time is the maximum allowed delay between the current time and the time at which a particular portion of content data must be sent in order to meet the real-time delivery requirements for content. If the delay for a particular portion of content data exceeds the max-lag-time, then an HTTP server endpoint must drop it instead of sending it. The value "*" indicates that the content data bytes will never expire. This guarantees that an HTTP client endpoint will receive all of the content data bytes.

The duration must be given in seconds.

Example:

- `realTimeInfo.dlna.org:`
DLNA.ORG_TLAG=1.75 `realTimeInfo.dlna.org:` DLNA.ORG_TLAG=*

M	A	DMS DMP DMR +PU+ +DN+ +PR1+ +PR2+	M-DMS M-DMP M- DMD	MIU	[39]

The max-lag-time value provides information to an HTTP client endpoint (for example, DMP) on how the content will be delivered. When it is infinite, all requested bytes of the original content will be delivered without any modifications (i.e., data loss). Otherwise, an HTTP client endpoint (for example, DMP) must not expect to receive the content unmodified (i.e. may contain some data loss).

In the case for real-time delivery of content, the max-lag-time negotiation aids an HTTP client endpoint (for example, DMP) to adjust its pre-buffering time and for an HTTP server endpoint (for example, DMS) its delay buffer. This negotiation aids an HTTP server endpoint from sending stale content data bytes instead of up-to-date ones.

7.5.5.3.19 MT HTTP media operations support

7.5.5.3.19.1 If a media operation (play, stop, pause/release, seek, scan operations) is supported for a content binary, content receivers must support the media operation for the entire known length of the content binary.

M	A	DMP DMR +DN+	M-DMP M-DMD	MIU	n/a

7.5.5.3.19.2 If an op-param (7.4.1.3.18.1) capability (for play, stop, pause/release, seek, scan operations) is supported for a content binary, content sources must support the media operation for the entire length of the content binary.

M	A	DMS +PU+	M-DMS	n/a	IEC 62481-2

In the case of time seek range, HTTP server endpoints are still allowed to align responses to frame boundaries as specified in MPEG-2 AV format, recommended decoder friendly alignment position, profiles: MPEG_PS_NTSC and MPEG_PS_PAL, 9.2.7.1 in IEC 62481-2.

7.5.5.4 HTTP transport – Interactive transfer guidelines

7.5.5.4.1 MT HTTP interactive transfer initiation

7.5.5.4.1.1 An interactive transfer must be initiated by the HTTP client endpoint using the HTTP GET method when using the HTTP transport protocol for media transfer.

M	A	DMPr DMP DMR	M-DMP	MIU	[39]

Interactive transfers with GET are used to get Image content for immediate rendering

7.5.5.4.1.2 An HTTP client endpoint requesting an interactive transfer must set the transferMode.dlna.org HTTP header to a value of "interactive" in the GET, or HEAD request to the server

M	A	DMPr DMP DMR	M-DMP	MIU	[39]

DLNA endpoint clients will still request interactive transfers of images without using the transferMode.dlna.org HTTP header. Subclause 7.5.5.2.33.2 requires that servers that receive such a request treat it as equivalent to a request for an interactive transfer for image media classes.

7.5.5.4.1.3 If an HTTP server endpoint supports interactive transfer for a particular content binary then the endpoint must respond to HTTP HEAD or GET requests with the transferMode.dlna.org HTTP header to indicate the server's interactive transfer capabilities for the specified URI.

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

It is good practice for servers to inform clients of their capabilities when queried with the HEAD request.

7.5.5.4.1.4 If an HTTP server endpoint receives an HTTP GET or HEAD request with a transferMode.dlna.org HTTP header value of "interactive" and the requested URI does not support interactive transfer of this content, then it must respond with an error code of 400 (bad request).

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.4.2 MT interactive transfer headers interactions

7.5.5.4.2.1 An HTTP client endpoint must not use the following headers when requesting an interactive transfer:

- TimeSeekRange.dlna.org;
- PlaySpeed.dlna.org;
- realTimeInfo.dlna.org.

M	A	DMPr DMP DMR	M-DMP	MIU	[39]

See 7.5.5.2.24 and 7.5.5.3.16 for more information on the operations of these headers

7.5.5.4.2.2 An HTTP server endpoint receiving the following headers as part of an interactive transfer must respond with the HTTP response code 400 (bad request):

- TimeSeekRange.dlna.org;
- PlaySpeed.dlna.org;
- realTimeInfo.dlna.org.

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.4.3 MT range behaviour for interactive transferred content

The range HTTP header may be used for interactive transfers when the "full random access data availability" model or the "limited random access data availability" model with mode-flag=1 applies to the HTTP server endpoint serving the content binary.

O	C	DMS DMPr DMR DMP +PU+ +PR1+ +PR2+	M-DMS M-DMP	MIU	[39]

The behaviour and usage model for the range header are governed by other guidelines. Specifically:

- 7.4.1.3.18 MM op-param describes how the op-param indicates whether the range header is supported under the "full random access data availability" model;
- 7.4.1.3.28 MM lop-npt and lop bytes: Common describes how lop-bytes indicates whether the range header is supported under the "limited random access data availability" model;
- 7.5.5.2.18 MT HTTP common random access data availability
- 7.5.5.2.19 MT HTTP – Data range of "full random access data availability";

7.5.5.2.20 MT HTTP – Data range of "limited random access data availability" (MT)

explain details about both models.

Interactive transfers involving stored content generally fall under the category of "full random access data availability" model. In some cases, an interactive transfer involving converted content will fall under the "limited random access data availability" model when the mode-flag is 1.

7.5.5.5 HTTP transport – Background transfer guidelines

7.5.5.5.1 MT HTTP background transfer initiation

7.5.5.5.1.1 A background transfer may be initiated by the HTTP client endpoint using the HTTP GET method when using the HTTP transport protocol for media transfer.

O	A	DMP+ +DN+	M-DMD	MIU	[39]

Background transfers with GET are used to download content from a media server

7.5.5.5.1.2 A background transfer may be initiated by the HTTP client endpoint using the HTTP POST method when using the HTTP transport protocol for media transfer.

O	A	+UP+	M-DMU	MIU	[39]

Background transfers with POST are used to upload content to a media server.

7.5.5.5.1.3 An HTTP client endpoint requesting a background transfer must set the transferMode.dlna.org HTTP header to a value of "background" in the GET, POST, or HEAD request to the server.

M	A	DMP+ +DN+ +UP+	M-DMD M-DMU	MIU	[39]

7.5.5.5.1.4 If an HTTP server endpoint supports background transfer for a particular content binary then the endpoint must respond to HTTP HEAD or GET requests with the transferMode.dlna.org:Background HTTP header and value to indicate the server's background transfer capabilities for the specified URI.

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

It is good practice for servers to inform clients of their capabilities when queried with the HEAD request.

7.5.5.5.1.5 If an HTTP server endpoint receives an HTTP GET or HEAD request with a transferMode.dlna.org HTTP header value of "background" and the requested URI does not support the background transfer mode, then it must respond with an error code of 400 (bad request).

M	A	DMS +PU+	M-DMS	n/a	[39]

7.5.5.5.2 MT background transfer header interactions

7.5.5.5.2.1 An HTTP client endpoint must not use the following headers when requesting a background transfer with either the GET or POST methods:

- TimeSeekRange.dlna.org;
- PlaySpeed.dlna.org;
- realTimeInfo.dlna.org.

M	A	DMP+ +DN+ +UP+	M-DMD M-DMU	MIU	[39]

See 7.5.5.2.24 and 7.5.5.3.16 for more information on the operations of these headers.

7.5.5.5.2.2 An HTTP server endpoint receiving the following headers as part of a background transfer must respond with the HTTP response code 400 (bad request):

- TimeSeekRange.dlna.org;
- PlaySpeed.dlna.org;
- realTimeInfo.dlna.org.

M	A	DMS +PU+ +PR1+ +PR2+	M-DMS	n/a	[39]

7.5.5.5.3 MT range behaviour for background transferred content

The range HTTP header may be used for background transfers when the "full random access data availability" model or the "limited random access data availability" model with mode-flag=1 applies to the HTTP server endpoint serving the content binary.

O	C	DMS DMP+ +PU+ +PR1+ +PR2+ +DN+ +UP+	M-DMS M-DMD M-DMU	MIU	[39]

The behaviour and usage model for the range header are governed by other guidelines. Specifically:

7.4.1.3.18 MM op-param describes how the op-param indicates whether the range header is supported under the "full random access data availability" model;

7.4.1.3.28 MM lop-npt and lop-bytes: Common describes how lop-bytes indicates whether the range header is supported under the "limited random access data availability" model;

7.5.5.2.18 MT HTTP common random access data availability

7.5.5.2.19 MT HTTP – Data range of "full random access data availability";

7.5.5.2.20 MT HTTP – Data range of "limited random access data availability" (MT)

explain details about both models.

Background transfers involving stored content generally fall under the category of "full random access data availability" model. In some cases, a background transfer involving converted content will fall under the "limited random access data availability" model when the mode-flag is 1.

7.5.5.6 HTTP transport – POST guidelines

7.5.5.6.1 MT content management HTTP POST content acquisition

7.5.5.6.1.1 An upload controller, M-DMU, or the MIU must implement a content transfer process by using an HTTP/1.1 POST request.

M	L	+UP+	M-DMU	MIU	[39]

See 7.4.1.7.26 MM/CM: Content transfer process for more information.

7.5.5.6.1.2 HTTP client endpoints must not use HTTP POST with an HTTP/1.0 indicator.

M	L	+UP+	M-DMU	MIU	[39]

HTTP POST is defined only for HTTP/1.1.

7.5.5.6.1.3 An upload controller, M-DMU, or the MIU requesting a content transfer process by using the HTTP/1.1 POST request, must set the transferMode.dlna.org HTTP header to a value of "background" in the request.

M	L	+UP+	M-DMU	MIU	[39]

7.5.5.6.1.4 A DMS or M-DMS that implements the upload device option (i.e., the upload AnyContainer) as indicated by the <dlna:X_DLNACAP> element described in 7.4.1.7.4.1), then it must accept HTTP POST requests for receiving content in content transfer operation.

M	R	DMS	M-DMS	n/a	[39]

This is required by the ContentDirectory specification as the baseline manner of transferring content to a UPnP AV MediaServer.

7.5.5.6.1.5 The HTTP client endpoint must issue its first HTTP POST request (to upload content) within 30 s of the CDS:CreateObject response.

M	L	+UP+	M-DMU	MIU	[39]

See 7.4.1.7.26 MM/CM: Content Transfer Process for more information.

7.5.5.6.1.6 If an HTTP client endpoint is using a persistent connection to perform multiple content transfer processes, then the HTTP client should issue subsequent HTTP POST requests within 5 min of the previous HTTP POST request's completion.

S	C	+UP+	M-DMU	MIU	[39]

M-DMUs or upload controllers should use persistent HTTP connections to upload multiple content binaries. Ideally, these content transfers happen with little delay between them.

However, as a general recommendation HTTP clients should not exceed 5 min of inactivity between transfers.

7.5.5.6.1.7 The HTTP server endpoint that is receiving an HTTP POST request is free to terminate the connection at any time by closing the TCP connection.

O	C	DMS	M-DMS	n/a	[39]

DMS or M-DMS devices implement this behaviour to indicate an error during the content transfer. For example, if the DMS no longer has enough space, it can terminate the TCP connection.

A DMS that does a TCP disconnection may automatically destroy CDS objects or <res> elements. However, if possible and appropriate, the DMS should allow an upload controller to retry a content transfer process, by not destroying the CDS object or <res> element.

7.5.5.6.1.8 The HTTP server endpoint that supports HTTP POST requests must support requests that are encoded with chunked transfer coding.

M	R	DMS	M-DMS	n/a	[39]

HTTP clients can use either the default HTTP message encoding or chunked transfer coding.

7.5.5.6.1.9 The HTTP client endpoint must provide the EXPECT HTTP header field with the value of "100-continue" in the request.

M	A	+UP+	M-DMU	MIU	[39]

Subclause 8.2.3 in [39] defines the behaviour when a HTTP client uses "Expect: 100-continue" in the request.

The HTTP client waits to send POST message body, i.e., a content binary, until it receives a response to the request.

7.5.5.6.1.10 If an HTTP server endpoint receives a POST request without the "EXPECT: 100-continue", then it should return an HTTP error of 400 (bad request) and terminate the TCP connection.

S	A	DMS	M-DMS	n/a	[39]

7.5.5.6.1.11 If an HTTP server endpoint that receives a POST request's HTTP headers and the HTTP server cannot accept the POST request's message body, then the HTTP server endpoint must not return an HTTP status code of 100 (continue).

M	R	DMS	M-DMS	n/a	[39]

Subclauses 7.5.5.6.1.13. and 7.5.5.6.1.14 provide examples of error cases.

7.5.5.6.1.12 An HTTP client endpoint that uses chunked transfer coding must use a zero-length chunk to indicate the end of the content binary.

M	A	+UP+	M-DMU	MIU	[39]

DMS and M-DMS devices look for the zero-length chunk to know if the content binary was completely sent.

7.5.5.6.1.13 If the HTTP server endpoint cannot accept an HTTP POST request due to the processing capacity or current state of the device, then the HTTP server endpoint should respond with an HTTP status code of 503 (service unavailable).

S	R	DMS	M-DMS	n/a	[39]

7.5.5.6.1.14 If the HTTP server endpoint cannot accept an HTTP POST request due to the lack of storage capacity, then the HTTP server endpoint should respond with an HTTP status code 507 (insufficient storage).

S	A	DMS	M-DMS	n/a	[39]

This error code aligns with RFC2518. DLNA does not use WebDAV as a normative reference.

7.5.5.6.1.15 If the HTTP server endpoint receives an entire entity body, then the HTTP server endpoint must respond with either an HTTP status code of 200 (OK) or an HTTP status code of 201 (created).

M	A	DMS	M-DMS	n/a	[39]

7.5.5.6.1.16 If an HTTP server endpoint rejects a POST entity body in the middle of data transfer (due to the processing capacity, current status of the device, storage capacity, etc.), then the HTTP server may terminate the TCP connection.

O	R	DMS	M-DMS	n/a	[39]

Vendors can determine whether or not data is actually stored.

7.5.5.6.1.17 If an HTTP server endpoint rejects a POST entity body in the middle of data transfer, then it should also send an HTTP error response right before terminating the TCP connection.

S	A	DMS	M-DMS	n/a	[39]

HTTP error responses allow the HTTP server to provide more details on the cause of the error. It is imperative that the HTTP server close the TCP connection after sending the error

response because the error can interfere with the logic associated with pipelined HTTP requests.

The requirement for HTTP clients to be tolerant of these scenarios means that the HTTP client can gracefully handle the TCP disconnect (i.e., devices that crash as a result of a TCP disconnect exhibit bad behaviour). There is no requirement that the HTTP client make the HTTP error response available to the user.

HTTP clients can use pipelined POST requests, but these guidelines do not recommend their usage because the upload system usage model favours a model where an upload controller uses a set of CDS:CreateObject and HTTP POST transactions, rather than a set of CDS:CreateObject requests followed by a set of HTTP POST transactions.

Lastly, clients should terminate the connection after receiving the POST response from the server because HTTP/1.1 POST transactions operate under persistent connection rules.

7.5.5.6.1.18 If an HTTP server endpoint rejects a POST entity body in the middle of data transfer by sending an HTTP error response, then it must terminate the TCP connection after sending the error response.

M	A	DMS	M-DMS	n/a	[39]

7.5.5.6.1.19 HTTP client endpoints must be tolerant of a TCP layer disconnect during an HTTP POST transaction, including those scenarios where the HTTP server endpoint sends an HTTP error response before the HTTP client endpoint has finished transmitting the POST request's message-body.

M	L	+UP+	M-DMU	MIU	[39]

7.5.5.6.1.20 If the HTTP server endpoint detects or initiates a TCP disconnect during a content transfer process that does not support the resume content transfer option, then the MediaServer must be capable of accepting the same upload AnyContainer or OCM: upload content request that created the CDS object, which has the failed content transfer.

M	A	DMS	M-DMS	MIU	n/a

In other words, the DMS should not return an UPnP error in response to a new CDS:CreateObject request when the upload controller attempts to retry a failed upload AnyContainer or OCM: upload content operation.

A DMS can implement a variety of behaviours to comply with this guideline:

- the DMS can immediately destroy the CDS object associated with the failed content transfer;
- the DMS can leave the CDS object in the CDS hierarchy for 30 min or less, from the point of the failure. The CDS:CreateObject response returns a new CDS object;
- the DMS can leave the CDS object in the CDS hierarchy for 30 min or less, from the point of the failure. The CDS:CreateObject response returns the same CDS object because the metadata specified in the request is exactly the same.

7.5.5.6.1.21 When the HTTP client endpoint detects a TCP disconnection before receiving the final response to the HTTP POST request, it must not assume that the HTTP server endpoint will store the transferred portion of the entity body. Specifically, this means the following.

- If the HTTP client endpoint wants to retry an upload process without using the resume content transfer or retry IFO attempt features, it must start completely over by doing the CDS:CreateObject request (as part of the upload AnyContainer or OCM: upload content operation) and following up with a new content transfer process. (It should be noted that retry IFO attempt applies only to the transfer of IFO files and is only available when resume content transfer is also available.)
- if the HTTP client endpoint wants to use the resume content transfer operation, then it must specify a first-byte-pos of res@dlna:uploadedSize for the Content-Range header in an HTTP POST request. In this case, CDS:CreateObject is not invoked.

M	C	+UP+	M-DMU	MIU	[39]

If resume content transfer is not supported and the HTTP client attempts to retry the HTTP POST transaction without invoking CDS:CreateObject again, then the retry attempt can fail. This type of a retry attempt is not covered by the DLNA guidelines.

7.5.5.6.2 MT HTTP content transfer error detection during HTTP POST

7.5.5.6.2.1 If an HTTP server endpoint observes a TCP disconnect under all of the following conditions, then the HTTP server must assume that the HTTP client endpoint failed to transfer the content:

- HTTP POST request does not use chunked transfer coding,
- the number of received bytes is less than the specified Content-Length.

If an HTTP server endpoint detects that the HTTP client endpoint failed to transfer the content and more than 35 min have elapsed since the end of the failed transfer attempt, then the MediaServer is obligated by 7.4.1.7 28. to automatically destroy the created CDS object.

M	A	DMS	M-DMS	n/a	[39]

HTTP client endpoints are required to use the Content-Length in HTTP POST requests that do not use chunked transfer coding.

Subclause 7.5.5.2.15 clarifies the uses of Content-Length for media transfers.

7.5.5.6.2.2 An HTTP server endpoint receiving a POST request must observe at least 5 min of data inactivity before closing the TCP connection and assuming that the HTTP client endpoint failed to transfer the content.

Data inactivity is defined as the HTTP server endpoint not receiving content data from the HTTP client endpoint even though there is an established TCP connection.

This guideline works in conjunction with 7.5.5.6.1.7 because this guideline assumes ideal network conditions and also assumes that neither user-initiated causes nor out-of-band system events cause the connection to be closed.

M	A	DMS	M-DMS	n/a	[39]

These guidelines specifies a 5 min timeout for stalled content transfer.

It should be noted that 7.5.5.6.1.7 permits HTTP servers to close the connection at any time.

7.5.5.6.3 MT client Content-Range

7.5.5.6.3.1 An HTTP client endpoint that uses the resume content transfer operation must specify the Content-Range HTTP header field in a POST request to specify the range of content data sent in the request.

M	A	+UP+	M-DMU	MIU	[39]

The Content-Range header is defined in 14.16 of RFC 2616.

7.5.5.6.3.2 An HTTP client endpoint that uses the resume content transfer operation must include and specify the instance-length parameter on the Content-Range HTTP header and last-byte position on the Content-Range HTTP header must be instance-length-1.

M	L	+UP+	M-DMU	MIU	[39]

The instance-length parameter (see 14.16 of RFC 2616) specifies the total size of the object being uploaded.

The instance-length parameter must specify valid value, so "*" must not be used for the instance-length parameter.

This guideline applies even when the HTTP client uses chunked transfer coding.

7.5.5.6.3.3 If an HTTP client endpoint specifies the Content-Range HTTP header the first-byte-pos must be equal to the content length already stored in a peer HTTP server.

M	A	+UP+	M-DMU	MIU	[39]

Length of already stored data (on a peer HTTP server) can be known through the res@dlna:uploadedSize in a CDS:Browse response.

The first-byte-pos parameter is defined in 14.16 of RFC 2616.

7.5.5.6.3.4 An HTTP client must use the Content-Range HTTP header only with POST requests that are part of a resume content transfer operation (i.e., use with GET or HEAD request is prohibited).

M	A	+UP+	M-DMU	MIU	[39]

7.5.5.6.3.5 An HTTP client endpoint must not specify the Content-Range HTTP header for a POST request addressed to res@importUri included in an object that does not support the resume content transfer functionality.

M	A	+UP+	M-DMU	MIU	[39]

Upload controller, MIU, or M-DMU is able to know if the MediaServer supports resume functionality by res@dlna:resumeUpload in CDS:CreateObject.

7.5.5.6.3.6 An HTTP client endpoint must not specify the Content-Range HTTP header for a POST request addressed to res@dlna:importIfoFileURI.

M	A	+UP+	M-DMU	MIU	[39]

The resume content transfer operation may be supported for res@importUri and cannot be used for res@dlna:importIfoFileURI. To recover from a failed transfer of an IFO file, the HTTP client simply does a retry IFO attempt, which is an HTTP POST request without the Content-Range HTTP header. It should be noted that type of transfer recovery is out of scope for res@importUri values.

7.5.5.6.4 MT server receiving Content-Range

7.5.5.6.4.1 An HTTP server endpoint that has sent a CDS:Browse response with res@dlna:uploadedSize must accept a POST request with Content-Range where first-byte-pos value is equal to the value of res@dlna:uploadedSize and appends its body to data that is already uploaded.

M	A	DMS	M-DMS	n/a	[39]

The Content-Range header and the first-byte-pos parameter are defined in 14.16 of RFC 2616.

7.5.5.6.4.2 If an HTTP server endpoint receives a POST request with Content-Range addressed to res@importUri and it does not support resume functionality at least for the specified URI in the request, it must respond an HTTP response with a status code of 406 (not acceptable).

M	A	DMS	M-DMS	n/a	[39]

7.5.5.6.4.3 If an HTTP server endpoint has resume functionality and receives a POST request with Content-Range addressed to res@importUri which includes a syntax error, it should respond an HTTP response with a status code of 400 (bad request).

S	A	DMS	M-DMS	n/a	[39]

If an instance-length is "*", in the Content-Range, DMS should respond with an HTTP response with a status code of 400.

7.5.5.6.4.4 If an HTTP server endpoint has resumed functionality and receives a POST request with the Content-Range header in which the content-range-spec has a first-byte-pos value that is equal to the object byte size already stored, then the HTTP server endpoint must append the incoming uploaded data to the stored one.

M	A	DMS	M-DMS	n/a	[39]

7.5.5.6.4.5 If an HTTP server endpoint has resume functionality and receives a POST request with the Content-Range header in which the content-range-spec has a first-byte-pos

value that is not equal to the object byte size already stored, then the HTTP server must send an HTTP response with a status code of 409 (conflict).

This guideline is applied only when another error condition is not satisfied.

M	A	DMS	M-DMS	n/a	[39]

7.5.5.6.5 MT HTTP POST pipeline

7.5.5.6.5.1 If an HTTP client endpoint initiates pipelined HTTP POST transactions, a subsequent pipelined POST request must happen after the previous message-body has been sent completely.

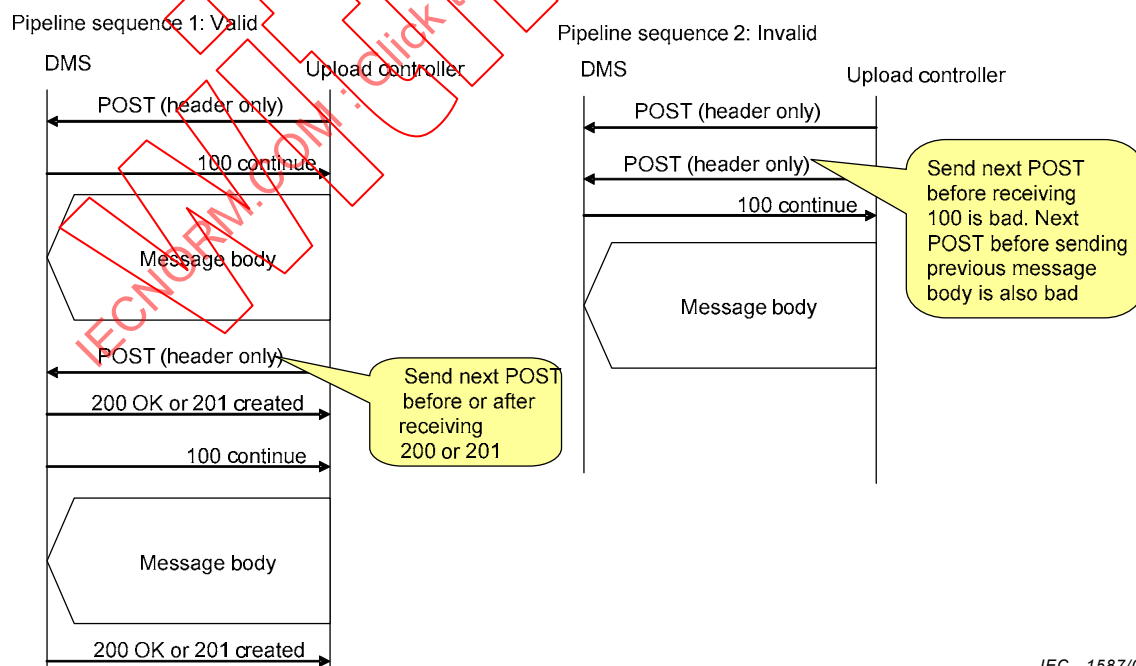
M	C	+UP+	M-DMU	MIU	[39]

An upload controller that starts a subsequent POST transaction before having sent the message-body for the previous POST transaction will confuse the HTTP Server Endpoint into thinking the subsequent POST transaction is the message-body for the first POST transaction.

7.5.5.6.5.2 An HTTP server endpoint may terminate the HTTP session after responding to a POST request with the 200 (OK) or 201 (created) responses.

O	C	DMS	M-DMS	n/a	[39]

DMS devices are not required to support pipelined HTTP POST transactions. Therefore, it remains the responsibility of the upload controller to retry an attempted pipelined POST transactions when the HTTP server does not support pipelined POST transactions.



IEC 1587/07

Figure 19 – Example of a valid and invalid pipelined POST transaction

7.5.6 RTP media transport

7.5.6.1 General

The real-time transport protocol (RTP) [44] together with its companion protocol (RTCP) constitutes the basis of a transport mechanism for real-time media streams. In DLNA, RTP is used in combination with the RTSP protocol [33], SDP protocol [34], RTP payload formats and their associated media format profiles. These protocols define the DLNA RTP media transport which is optionally supported in DLNA, in addition to the required HTTP transport.

Subclause 7.5.6 defines guidelines for the implementation of the RTP media transport in the context of home networking. These apply to the playback of DLNA content and the streaming transactions between DLNA device classes. These guidelines do not specify behaviour for non-DLNA device entities. A device class may be implemented by software running on a more general-purpose device/platform. For example, the RTP server of a serving endpoint may be used to serve DLNA and non-DLNA content. These guidelines would apply only when the DLNA-content is being served.

For these guidelines, RTP/RTCP is a protocol which is carried over UDP/IPv4 for unicast connections. A graphical representation of the reference protocol stack for the RTP media transport is shown in Annex H. The RTP media transport is enriched with the support of advanced features. For example, RTP retransmission can be used to increase reliability of media delivery. The RTP media transport also supports mechanisms for adaptive media delivery to improve continuous playback even in adverse network conditions. These guidelines also cover the RTSP protocol, which is carried over TCP connections. RTSP supports pause, seek and scan media operations (fast forward, slow forward, fast backward and slow backward).

Lastly, the guidelines do not specify interoperability for the transfer of content for either the upload system usage or the download system usage. Likewise, the guidelines also assume that the RTP media transport applies only to the streaming transfer mode. The interactive and background transfer modes do not apply to the RTP media transport.

The guidelines for RTP are organized as follows.

- RTP/RTCP Protocols: RTP media transport – RTP/RTCP (7.5.6.2.1 to 7.5.6.2.18) provides general RTP media transport guidelines. RTP serving endpoint (7.5.6.3.1 to 7.5.6.3.23) provides RTP Serving Endpoints guidelines, while RTP receiving endpoint (7.5.6.4.1 to 7.5.6.4.16) provides RTP guidelines for receiving endpoints. These subclauses also include guidelines for RTCP.
- Adaptation of media format profiles 7.5.7.2.1 through 7.5.7.11.6 provides guidelines on adapting several media format profiles for use with RTP and RTP payload formats.
- RTSP/SDP protocols: RTSP and SDP signalling for DLNA (7.5.8.2.1 to 7.5.8.2.80) includes guidelines for the support of the real-time streaming protocol [33] and the support for the session description protocol [34].

It should be noted that the terms serving endpoint and receiving endpoint are specific to RTP5, and they are defined specifically in 7.5.6.2.3 and 7.5.6.2.4. Respectively, they represent the different RTP, RTCP, RTSP, and SDP server/client components needed for a content source or content receiver using RTP.

⁵ In the previous version of the DLNA guidelines, serving endpoint and rendering endpoint were used. Since the system usages for these guidelines is greatly expanded, those terms were deemed insufficient to represent the different content roles on the network.

7.5.6.2 RTP media transport – RTP/RTCP protocols

7.5.6.2.1 MT RTP optional transport – RTP

DLNA devices may implement RTP over UDP as an optional media transport. Guidelines in 7.5.6.2.1 to 7.5.8.2.80 only when the RTP media transport is implemented.

O	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	n/a

RTP is an optional media transport. If RTP is optionally implemented by serving endpoints and receiving endpoints, the guidelines that follow in these tables must be implemented as described.

RTP is recommended for certain applications, for example, streaming transfer of live content.

7.5.6.2.2 MT RTP applicable media class

Serving endpoints and receiving endpoints using this media transport must use it only for audio or audio/video media classes

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	n/a

7.5.6.2.3 MT RTP on serving endpoints

Serving endpoints must comply with required features of RFC 3550 with constraints and additions as specified in RTP serving endpoint (7.5.6.3.1 to 7.5.6.3.23).

M	A	DMS +PU+	M-DMS	n/a	[44]

Serving endpoints in DLNA have no need to receive RTP packets.

7.5.6.2.4 MT RTP on receiving endpoints

Receiving endpoints must comply with required features of [44] (RFC 3550) with constraints and additions as specified in RTP receiving endpoint (7.5.6.4.1 to 7.5.6.4.16).

M	A	DMP DMR	M-DMP	MIU	[44]

Receiving endpoints in DLNA have no need to transmit RTP packets.

7.5.6.2.5 MT RTP RTSP/SDP

Serving endpoints and receiving endpoints must use RTSP and SDP as defined in RTSP and SDP signalling for DLNA (7.5.8.2.1 to 7.5.8.2.80).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	n/a

Certain RTP session parameters, such as UDP port, RTP clock frequency and RTP payload type, must be communicated before RTP packets can be exchanged. These parameters must be communicated using RTSP and SDP as defined in RTSP and SDP signalling for DLNA (7.5.8.2.1 to 7.5.8.2.80).

7.5.6.2.6 MT RTP profile support

7.5.6.2.6.1 The following RTP profile must be supported by serving endpoints and receiving endpoints.

RTP profile for audio and video conferences with minimal control as defined in RFC 3551, also called RTP/AVP.

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44] [45]

7.5.6.2.6.2 The following RTP profile may optionally be supported by serving endpoints and receiving endpoints.

Extended audio/visual profile for RTCP based feedback as defined in [14] also called RTP/AVPF.

O	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[14]

7.5.6.2.7 MT RTP serving endpoint DLNA media format profile support

7.5.6.2.7.1 A serving endpoint must support at least one "RTP support level" as detailed below.

M	A	DMS +PU+	M-DMS	n/a	n/a

RTP support levels (A or B) enable better understanding of the extent to which serving endpoints support RTP.

7.5.6.2.7.2 If a serving endpoint claims "RTP support level A" it must support transmitting at least one media format profile over RTP.

M	A	DMS +PU+	M-DMS	n/a	n/a

This is a very basic requirement: an RTP-capable serving endpoint must be able to stream at least something over RTP (otherwise, it is simply not an RTP-capable serving endpoint).

7.5.6.2.7.3 If a serving endpoint claims "RTP support level B" for each media format profile that it supports transmitting over HTTP, it must also support transmitting that media format profile over RTP.

M	A	DMS +PU+	M-DMS	n/a	n/a

It should be noted that all devices of level B also satisfy level A.

This level of support avoids confusion by the end-user. If a level B serving endpoint supports a certain set of media formats and it supports RTP, then RTP is supported for all these media formats.

7.5.6.2.8 MT RTP receiving endpoint DLNA media format profile support

7.5.6.2.8.1 A receiving endpoint must support at least one "RTP support level", as detailed below.

M	A	DMP DMR	M-DMP	MIU	n/a

RTP support levels (A or B) enable better understanding of the extent to which a receiving endpoint supports RTP.

RTP support levels are not signaled between the serving endpoints and the receiving endpoints.

7.5.6.2.8.2 If a receiving endpoint claims "RTP support level A", it must support receiving at least one media format profile over RTP.

M	A	DMP DMR	M-DMP	MIU	n/a

This is a very basic requirement: an RTP-capable receiving endpoint must be able to receive at least something over RTP (otherwise, it is simply not an RTP-capable receiving endpoint).

7.5.6.2.8.3 If a receiving endpoint claims "RTP support level B" for each media format profile that it supports receiving over HTTP, it must also support receiving that media format profile over RTP.

M	A	DMP DMR	M-DMP	MIU	n/a

It should be noted that all devices of level B are also satisfying level A.

This level of support avoids confusion by the end-user. If a level B receiving endpoint supports a certain set of media formats and it supports RTP, then RTP is supported for all these media formats.

7.5.6.2.8.4 If a receiving endpoint claims "RTP support level B" and supports a program stream based media format profile (over RTP), then it must also support the corresponding elementary stream based media format profile for transport over RTP. In particular:

- if receiving endpoint supports MPEG_PS_PAL then it must also support MPEG_ES_PAL;
- if receiving endpoint supports MPEG_PS_NTSC then it must also support MPEG_ES_NTSC;
- if receiving endpoint supports MPEG_PS_PAL_XAC3 then it must also support MPEG_ES_PAL_XAC3;
- if receiving endpoint supports MPEG_PS_NTSC_XAC3 then it must also support MPEG_ES_NTSC_XAC3.

M	A	DMP DMR	M-DMP	MIU	n/a

This enables a serving endpoint to choose PS or ES encapsulation depending on application requirements.

It should be noted that the ES-based media format profiles are defined for RTP media transport only (not for HTTP media transport).

7.5.6.2.9 MT RTP payload type definitions

Serving endpoints must use the payload type(s) defined in A/V media format profiles in the context of RTP transmission (7.5.7.2.1 to 7.5.7.2.6) when transmitting DLNA media format profile content over RTP.

M	A	DMS +PU+	M-DMS	n/a	n/a

For each media format profile exactly one RTP encapsulation is defined, except for full TS with zero TTS. This implies that the RTP encapsulation can be inferred from the media format profile as carried in the 4th protocolInfo field except for full TS with zero TTS. Full TS with zero TTS encapsulation can be determined using a=rtmpmap in SDP.

It should be noted that transport stream and program stream based media format profiles are encapsulated as a single RTP stream, whereas other file formats (for example, MP4, 3GPP) are encapsulated as two separate RTP streams for AV.

7.5.6.2.10 MT RTP header timestamps during PLAY requests with speed or scale headers

7.5.6.2.10.1 A serving endpoint must maintain RTP header timestamp values that correspond to playback timing during scaled playback (i.e. a PLAY request with a scale header with a value not equal to 1) as defined in RFC 2326 Appendix B. The output of the serving endpoint during scaled playback must be compliant with the syntax rules of the media format profile. The values of the RTP header timestamps must be compliant with the requirements for the RTP payload format in use (for example, video/MP2P, video/MP2T, or video/MPV).

For example, a receiving endpoint requests a scale value of 2 for a PLAY request. The serving endpoint could comply by dropping every other video frame (or utilize some other algorithm) while maintaining the original frame rate of the content. Assuming a 90 KHz RTP header timestamp clock timebase and PS encapsulation, the RTP header timestamp values would increase by 90 000 for each second of playback, even though the NPT time reference would increase by 2,000 each second.

As a further example, if the scale value was – 4 (backwards at 4 times the nominal playback rate), the RTP header timestamp values will also increase by 90 000 for each second, but the NPT time reference would decrease by – 4,000 during the same time period.

M	C	DMS +PU+	M-DMS	n/a	[33]

This guideline clarifies the relationship between the RTP header timestamps and the temporal scale of the content being transmitted when the scale value is not equal to 1. (A scale value of 1 implies normal playback scale and direction.) The output produced by scaling is essentially a "normal" 1X sequence of RTP packets which would have RTP header timestamps that are equivalent to a non-scaled content stream. This should be true regardless of the algorithm used by the serving endpoint to generate the content.

For example, consider a serving endpoint A which implements a scale value of 2 by dropping every other frame in a 30 fps stream resulting in a content stream which still contains 30 frames per second, but which appears to be playing at twice the normal speed. Serving endpoint B should next be considered which implements the same scale value of 2 by sending 4 I-Frames a second with a presentation time of 250 ms per frame, which will also appear to be playing twice as fast. In both cases the RTP header timestamp values would increase by 90 000 each second.

7.5.6.2.10.2 A serving endpoint must not change the RTP header timestamp values that correspond to playback timing in response to a PLAY request with a speed header.

For example, a receiving endpoint requests a speed value of 2 for a PLAY request. The serving endpoint would transmit the content at twice the nominal playback rate while maintaining the nominal RTP header timestamp values within each packet.

M	L	DMS +PU+	M-DMS	n/a	[33]

It should be noted that this only applies to the RTP header timestamp and not to the wall clock as carried within the RTP extension header.

7.5.6.2.10.3 If a PLAY request includes both speed and scale headers, the serving endpoint must satisfy the parameters of the scale header, and then deliver the resultant sequence of RTP packets to the network as specified in the speed header.

M	C	DMS +PU+	M-DMS	n/a	[33]

This guideline clarifies the order in which operations should be carried out if both scale and speed headers are included in a PLAY request. Details on the scale and speed headers can be found in 7.5.8.2.41 through 7.5.8.2.44.

7.5.6.2.11 MT RTP/RTCP must use UDP transport

Serving endpoints and receiving endpoints must send and receive any RTP and RTCP packets as UDP packets.

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44]

7.5.6.2.12 MT RTP unicast support

Serving endpoints and receiving endpoints must support the transmission of RTP and RTCP packets to unicast IPv4 addresses.

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	n/a

7.5.6.2.13 MT RTP RTCP support required

RTCP must be implemented as specified in RFC 3550 and as specified by the RTP profile in use with the constraints defined in RTP media transport – RTP/RTCP (7.5.6.2.1 to 7.5.6.2.18), RTP serving endpoint (7.5.6.3.1 to 7.5.6.3.23), RTP receiving endpoint (7.5.6.4.1 to 7.5.6.4.16) and RTSP and SDP Signaling for DLNA (7.5.8.2.1 to 7.5.8.2.80).

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44]

RTCP is necessary for rate control and synchronization of RTP streams.

7.5.6.2.14 MT RTP/RTCP UDP port number usage

RTP media transport must be carried out over an even RTP/UDP port number (i.e. 2n). RTCP control must be carried out over the next incremented UDP port number (i.e., 2n+1)

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44]

7.5.6.2.15 MT RTP unknown RTCP packet types

Serving endpoints and receiving endpoints must tolerate unknown RTCP packet types.

Tolerate means that the endpoints must be able to parse and interpret or parse and ignore such packets.

M	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44]

7.5.6.2.16 MT RTP RTCP simplified report interval calculation

7.5.6.2.16.1 RTCP reports must be sent at the rate that is in accordance with the SDP provisions (if any).

M	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44]

If SDP specifies an RTCP reporting rate, then this rate has the highest precedence (see 7.5.8.2.70.)

7.5.6.2.16.2 In the absence of SDP provisions any RTCP reports must be generated and sent at the rate that is in accordance with one of the following:

- the RTP profile (AVP or AVPF) in use;
- once every 5 s randomized within the interval [0,5, 1,5] times, such that the resulting RTCP transmission interval is a random number in the interval [2,5, 7,5]s.

M	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44]

The rules for calculating RTCP report intervals in RFC 3550 are complex to accommodate large multicast sessions.

7.5.6.2.17 MT RTP version number

Serving endpoints and receiving endpoints must set the version number in the RTP header to be 2. Serving endpoints and receiving endpoints must accept RTP packets with versions of 2.

M	R	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[44]

7.5.6.2.18 MT RTP DLNAQOS RTCP traffic

7.5.6.2.18.1 If DLNAQOS as defined in 7.2 is implemented, all RTCP messages generated by a serving endpoint must be tagged with DLNAQOS_2, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), in accordance with Table 8.

M	R	DMS +PU+	M-DMS	n/a	n/a

Because RTP is defined only for UDP, RTCP messages are not subject to the same constraints as HTTP messages sent over TCP. RTCP requests do not have to have the same priority that the server will use to deliver the content.

RTCP packets from a DMS will contain sender report (SR) messages when the server is streaming, or receiver report (RR) messages when the server is idle. The SR and RR messages are important, but arguably not any more important than the RTP packets because if the network is experiencing congestion, then the RTP stream has more serious problems than any information lost in these messages (for example, clock sync).

7.5.6.2.18.2 If DLNAQOS as defined in 7.2 is implemented, all RTCP messages generated by a receiving endpoint must be tagged with DLNAQOS_3, or a lower DLNAQOS_UP value (where "or a lower" is defined by 7.2.4.2.2.2 and 7.2.4.2.2.3), in accordance with Table 8.

M	R	DMP DMR	M-DMP	MIU	n/a

All feedback messages are time critical, and especially important at times when the DLNAQOS_2 RTP traffic is suffering from congestion.

RTCP messages sent by a DMP include receiver report (RR) messages and RTCP payload types of 205 and 206. In the RTP/AVPF profile, RTCP payload type 205 is defined as a transport-layer feedback message and 206 is defined as a payload-specific feedback message. This will cover not only RTCP NACK messages, but also other kinds of RTCP-based feedback that we may want to add in the future.

RTCP RR messages contain statistics about lost RTP packets, so they are time critical if the server makes any decisions based on those statistics. Also, all RTCP packets that the client sends must include a RR message. Even if the client just wants to send a NACK, the RTCP packet must also include an RR message.

7.5.6.3 RTP serving endpoint requirements

7.5.6.3.1 MT RTP timestamp offset

7.5.6.3.1.1 All RTP timestamp values should have an offset component which is a fixed 32-bit value which is added to the RTP clock value as each RTP timestamp is created.

S	C	DMS +PU+	M-DMS	n/a	[44]

7.5.6.3.1.2 The serving endpoint may select an offset value at the start of the RTP stream, and this value must be maintained throughout the RTP stream unless a timestamp discontinuity is indicated.

O	C	DMS +PU+	M-DMS	n/a	[44]

Some RTP payload formats may allow a timestamp discontinuity to be indicated using the "M" bit in the RTP packet header, or through other means.

7.5.6.3.2 MT RTP SSRC uniqueness – Serving endpoints

Serving endpoints must ignore SSRC collisions.

M	L	DMS +PU+	M-DMS	n/a	[44]

RTP was designed for large multicast conferencing-type of applications.

SSRC collisions can occur in large conferences where multiple receivers/senders choose their own SSRC.

In a unicast environment, the RTP serving endpoint must ignore SSRC collisions and continue the session with the negotiated SSRC values

7.5.6.3.3 MT RTP serving endpoint RTP retransmission support

A serving endpoint may retransmit RTP packets if it and the receiving endpoint both are using the RTP/AVPF profile.

O	A	DMS +PU+	M-DMS	n/a	[14]

7.5.6.3.4 MT RTP serving endpoint RTP retransmissions

7.5.6.3.4.1 A serving endpoint that retransmits RTP packets must use one of the following two retransmission methods.

- The retransmitted packet is formatted using the RTP payload format audio/rtx or video/rtx, in accordance with [16].
- The retransmitted packet is sent as an identical copy of the original packet and is sent to the same UDP port as the original RTP packet.

M	A	DMS +PU+	M-DMS	n/a	[16]

For interoperability reasons, it is suggested that the serving endpoint implement both transmission methods and chooses the one supported by the receiving endpoint as announced in 7.5.8.2.29.2.

Implementation suggestion. If serving endpoint implements a buffering scheme for retransmissions, any encoding rate adaptation techniques, such as bit stream switching, trans-rating and frame skipping, it is recommended not to flush the retransmission buffer, as it may be difficult to reconstruct the original data.

7.5.6.3.4.2 A serving endpoint that retransmits RTP packets should format the retransmitted packet using the RTP payload format audio/rtx or video/rtx, in accordance with [16].

S	A	DMS +PU+	M-DMS	n/a	[16]

7.5.6.3.4.3 A serving endpoint that retransmits RTP packets may send the retransmitted packet as an identical copy of the original packet and it is sent to the same UDP port as the original RTP packet.

O	A	DMS +PU+	M-DMS	n/a	[16]

7.5.6.3.5 MT RTP packet size

7.5.6.3.5.1 A serving endpoint should limit the size of the RTP packet so that the size of the entire packet including protocol headers will not exceed the maximum transmission unit (MTU) used in the home network.

S	A	DMS +PU+	M-DMS	n/a	[23]

The purpose of this recommendation is to avoid IP packet fragmentation and the performance penalty associated with this.

A serving endpoint can determine the maximum MTU size for a given connection dynamically using the algorithm described in RFC 1191. If the serving endpoint chooses not to, or is unable to determine the size dynamically, it can assume an MTU of size of 1492 bytes is safe.

7.5.6.3.5.2 A serving endpoint must not select an RTP packet size in excess of 4096 bytes.

M	A	DMS +PU+	M-DMS	n/a	[23]

This provides the receiving endpoint with an upper limit on the size of an incoming RTP packet.

7.5.6.3.6 MT RTP UDP port usage

7.5.6.3.6.1 If an RTP media format profile encapsulation (see A/V media format profiles in the context of RTP transmission, 7.5.7.2.1 to 7.5.7.2.6) mandates that content be sent as multiple RTP streams, the serving endpoint must support sending them to the same (2n, 2n+1) UDP port pair, even if the receiving endpoint requests them to be the same.

M	C	DMS +PU+	M-DMS	n/a	[44]

When RTSP is used, the receiving endpoint selects the UDP ports for each RTP stream, and although not recommended by [44], it may choose the same UDP port pair for multiple RTP streams.

7.5.6.3.6.2 If an RTP media format profile encapsulation (see A/V media format profiles in the context of RTP transmission, 7.5.7.2.1 to 7.5.7.2.6) mandates that content be sent as multiple RTP streams, the serving endpoint must support sending them to different (2n, 2n+1) UDP port pairs.

M	R	DMS +PU+	M-DMS	n/a	[44]

This results in each RTP stream being sent on a different RTP session, which is recommended by [44].

7.5.6.3.7 MT RTP RTCP first sender report

A serving endpoint should transmit a sender report as soon as possible after sending the first RTP packet of an RTP stream.

S	L	DMS +PU+	M-DMS	n/a	[44]

This allows receiving endpoints to synchronize RTP streams.

7.5.6.3.8 MT RTP required RTCP SDES items

The only RTCP SDES item required is CNAME.

M	C	DMS +PU+	M-DMS	n/a	[44]

The CNAME item is required according to 6.1 of [44].

7.5.6.3.9 MT RTP RTCP BYE packets recommended

Serving endpoints should send an RTCP BYE packet when leaving an RTP session.

S	L	DMS +PU+	M-DMS	n/a	[44]

7.5.6.3.10 MT RTP RTCP receiver reports tolerance

Serving endpoint must tolerate RTCP receiver reports.

M	C	DMS +PU+	M-DMS	n/a	[44]

7.5.6.3.11 MT RTP RTCP transmission interval in case of RTP translators

If a serving endpoint acts as an RTP translator, each received RTCP packet must be forwarded immediately upon packet arrival. The rule 7.5.6.2.16.2 must not be applied.

M	C	DMS +PU+	M-DMS	n/a	[44]

This rule is compliant to [44] (RTCP processing in translators)

7.5.6.3.12 MT RTP uniqueness of RTP SSRC

Each RTP stream must use a different SSRC.

M	A	DMS +PU+	M-DMS	n/a	[44]

This is especially helpful to the receiving endpoint if multiple RTP streams are sent to the same UDP port.

7.5.6.3.13 MT RTP buffer fullness report processing

7.5.6.3.13.1 The serving endpoint may use buffer fullness reports (BFRs) to compute the average rate of depletion of the receiving endpoint's buffer.

O	A	DMS +PU+	M-DMS	n/a	n/a

This information can be used to detect network congestion and transrate the media stream.

7.5.6.3.13.2 If BFRs indicate that the receiving endpoint's buffer levels are low, then the serving endpoint may temporarily increase the transmission rate to fill the receiving endpoint's buffer.

A low buffer level is defined as being below the target buffer duration value.

Temporarily increase the transmission rate means that the transmission rate increases until the serving endpoint gets a report indicating the buffer level is equal to, or greater than, the target buffer duration value.

O	A	DMS +PU+	M-DMS	n/a	n/a

This can be used for faster startup and to make the receiving endpoint more tolerant to network jitter.

7.5.6.3.13.3 If the serving endpoint changes the transmission rate in response to a BFR, RTP timestamps must not be adjusted to reflect the changed transmission rate.

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.6.3.14 MT RTP transmission rate adaptation

7.5.6.3.14.1 If the SETUP request included the Buffer-Info.dlna.org header (7.5.8.2.17), and the parameter "BFR=1" was specified on that header, then the serving endpoint may perform transmission rate adaptation.

O	A	DMS +PU+	M-DMS	n/a	n/a

7.5.6.3.14.2 If the serving endpoint has indicated that transmission rate adaptation is possible (7.5.8.2.79.1) and the receiving endpoint specified a target buffer duration for an RTP stream using the Buffer-Info.dlna.org header (guideline 0), then the serving endpoint may perform transmission rate adaptation for the first target buffer duration of NPT.

O	A	DMS +PU+	M-DMS	n/a	n/a

Example: If the SETUP request included "TD=5000" on the Buffer-Info.dlna.org header, the serving endpoint does not need to pace the data when transmitting the first 5 s worth of data.

7.5.6.3.15 MT RTP wall clock time samples

If a receiving endpoint requests the serving endpoint to add wall clock time samples using the RTSP header WCT.dlna.org, as described in 7.5.8.2.37.2, then it is strongly recommended for the serving endpoint to add wall clock time samples to RTP packets conforming to 7.5.6.3.16 through 7.5.6.3.21 below.

S	A	DMS +PU+	M-DMS	n/a	n/a

The wall clock time sample denotes the actual transmission time of the RTP packet very accurately.

This enables receiving endpoints to perform clock recovery to enable seamless A/V rendering.

It should be noted that it is up to the receiving endpoint whether it chooses to perform clock recovery or not. These guidelines just strongly recommend that the serving endpoint does the minimum necessary to make it possible.

For MPEG-2 TS or PS encapsulation, the RTP time stamps provide an alternate means for clock recovery. However, RTP time stamps only denote the actual moment of transmission as accurately as the pacing has been (i.e., 35 ms worst case), whereas the wall clock time sample is within 2,5 ms accurate. This enables more reliable clock recovery.

For encapsulation of elementary streams the RTP timestamps denote sample time and not transmission time, making them unsuitable for clock recovery.

7.5.6.3.16 MT RTP wall clock time sample source

7.5.6.3.16.1 The wall clock time samples in the RTP packets must be obtained from the wall clock time source that is used for generating the NTP timestamp in the RTCP SR messages.

M	A	DMS +PU+	M-DMS	n/a	[24] [44]

The RTCP SR contains a sample of the wall clock time – "NTP timestamp" – when the RTCP packet was sent, along with a sample of the clock generating the RTP timestamps for the RTP stream involved.

This means that the respective RTCP sender reports for the different RTP streams (for this media format) can be used to relate all RTP timestamps to the common wall clock time.

This can in turn be used for inter-media synchronization (i.e., "lip sync").

Subclauses 7.5.6.3.16 through 7.5.6.3.21 effectively enable reconstruction of the serving endpoint wall clock time at the receiving endpoint, and thereby serve the same role as, for example, NTP in more conventional RTP implementations.

7.5.6.3.16.2 The wall clock time source must be at least as accurate as specified by the system clock specification (if any) of the RTP payload content concerned.

M	A	DMS +PU+	M-DMS	n/a	n/a

For example, with MPEG-2 content the 90 KHz clock source must be accurate to within $\pm 30 \times 10^{-6}$ and have a slew rate of less than 0,075 Hz per second as defined in ISO/IEC 13818-1.

7.5.6.3.17 MT RTP wall clock time samples for all packets

If a serving endpoint adds wall clock time samples, then they must be added to each RTP packet of each RTP stream associated with the media format being served.

M	A	DMS +PU+	M-DMS	n/a	n/a

Sampling the wall clock time for each packet of each RTP stream results in the largest number of clock samples per second.

For proper clock reconstruction it is advantageous to have as many samples per second as possible.

7.5.6.3.18 MT RTP wall clock time sample accuracy

The wall clock time sample in an RTP packet must represent the moment that the packet is handed over to the network. The standard deviation of the distribution of the errors must be less than 2,5 ms.

M	A	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-9

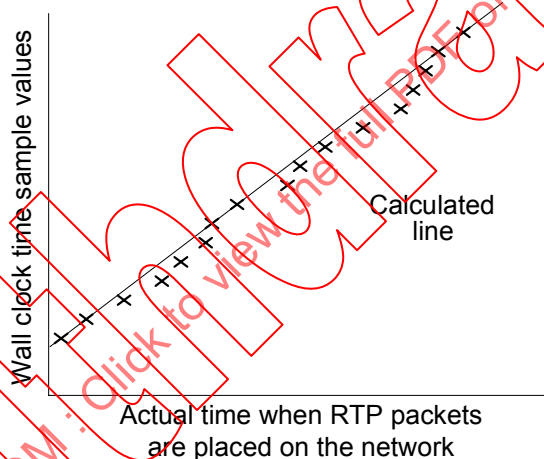
Puts upper bound on additional jitter imposed by the serving endpoint's implementation (i.e., "OS jitter"). Typically, network jitter will dominate over "OS jitter".

The RTP timestamps (Y axis) vs. the actual arrival time on the network (X axis) will be plotted against each other, and a line calculated using a (simple average/least mean squares) fitting algorithm. The slope of the calculated line will be unity if the clock source is accurate. A positive or negative slope will indicate a frequency error or "drift". The second derivative or "curve" of the line represents the slew rate of the clock source.

For testing suggestions, see footnote⁶.

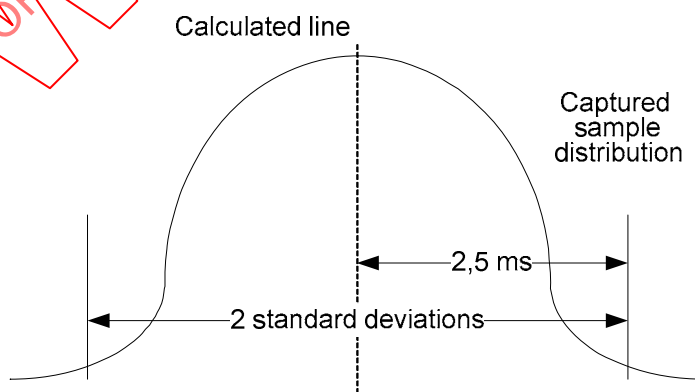
The accuracy of the wall clock time samples will be determined by comparing each wall clock time sample with the actual time the RTP packet is placed on the network as described in ISO/IEC 13818-9 over a period of 10 min. The slope of the calculated line (Figure 20) using a least mean squares will be used to determine accuracy of the wall clock time samples.

The distribution of the distance between the observed samples and the calculated line (Figure 20) must be such that two standard deviations of the distribution is $\pm 2,5$ ms as shown in Figure 21.



IEC 1588/07

Figure 20 – Calculated line



IEC 1589/07

⁶ This guideline should be tested by connecting a packet capturing device directly to the device under test (DUT) using a crossover Ethernet cable for 802.3 interfaces and/or an 802.11 station or access point in close proximity with no observable radio interference. This will allow the arrival time on the network to be measured as accurately as possible by minimizing the effect of network jitter. However, it should be noted that presence of network jitter does not invalidate the accuracy of this test with respect to clock source accuracy.

Figure 21 – Wall clock time sample accuracy distribution

7.5.6.3.19 MT RTP wall clock time sample unaffected by speed, scale, and BF

7.5.6.3.19.1 Wall clock time samples must represent the "actual" transmission time, irrespective of whether the nominal transmission rate is used not.

Transmission rates other than nominal may occur when

- a) a rate other than 1,0 is requested by the use of the speed header; or
- b) transmission rate adaptation is performed as described in 7.5.8.2.79.2 and 7.5.8.2.79.4.

M	C	DMS +PU+	M-DMS	n/a	n/a

The wall clock time sample will be unaffected by scale header, speed header, or BFR as it is not tied to the media stream in any way.

When the transmission rate is increased the wall clock time samples of adjacent packets will have smaller temporal distance compared to normal rate. When the transmission rate is decreased the temporal distances will increase.

7.5.6.3.19.2 Wall clock time samples must represent the "actual" transmission time, irrespective of whether the content is scaled (scale header not 1) or not.

M	C	DMS +PU+	M-DMS	n/a	n/a

7.5.6.3.20 MT RTP wall clock time sample representation

The middle 32 bits of the wall clock time sample must be stored in the RTP packet.

M	A	DMS +PU+	M-DMS	n/a	[24] [44]

Use of the middle 32 bits reduces packet header overhead and is more commonly used in [44]. It still features a resolution of about 65 KHz (same order of magnitude as 90 KHz RTP timestamps used for PS/TS encapsulation) and a wrap around time of about 18 h.

From RFC 3550 [44]: Wallclock time (absolute date and time) is represented using the timestamp format of the network time protocol (NTP), which is in seconds relative to 0 h UTC on 1 January 1900. The full resolution NTP timestamp is a 64-bit unsigned fixed-point number with the integer part in the first 32 bits and the fractional part in the last 32 bits. In some fields where a more compact representation is appropriate, only the middle 32 bits are used; that is, the low 16 bits of the integer part and the high 16 bits of the fractional part. The high 16 bits of the integer part must be determined independently.

7.5.6.3.21 MT RTP wall clock time sample RTP header extension

7.5.6.3.21.1 The 32-bit wall clock time sample in an RTP packet must be encoded by means of a header extension.

M	A	DMS +PU+	M-DMS	n/a	n/a

Headers extensions offer full backward compatibility: an implementation that does not recognize a header extension must silently ignore it.

7.5.6.3.21.2 The X bit in the RTP header must be set to "1" to indicate the presence of a header extension.

M	R	DMS +PU+	M-DMS	n/a	[44]

7.5.6.3.21.3 For RTP streams using the RTP/AVP profile the "defined by profile" field in the header extension header must be set to 0x2356 to uniquely define this DLNA "wall clock time sample" header extension.

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.6.3.21.4 For RTP streams using the RTP/AVPF profile the "defined by profile" field in the header extension header must be set to 0x2356 to uniquely define this DLNA "wall clock time sample" header extension.

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.6.3.21.5 If no additional header extensions (in addition to wall clock time sample) are required in the RTP packet header, the "length" field in the header extension header must be set to 1 to indicate a single 32-bit word in the header extension contents.

M	A	DMS +PU+	M-DMS	n/a	n/a

To cater to future needs, optionally, another header extension may follow the wall clock time sample header extension. In this case, the "length" field must indicate the total length of the header extension(s) following the header of the wall clock time sample header extension. It should be noted that future DLNA guidelines may apply this rule recursively. The case of a single wall clock time sample header extension is illustrated in Figure 22. The case of another header extension following it is illustrated in Figure 23.

7.5.6.3.21.6 Another RTP header extension may follow the wall clock time sample header extension, as described in [44].

O	R	DMS +PU+	M-DMS	n/a	[44]

7.5.6.3.21.7 If another header extension follows the wall clock time sample header extension, the "length" field in the wall clock time sample header extension header must be set to the total size of the header extension following it (including its header) plus 1 to indicate the single 32-bit word in the wall clock time sample header extension contents.

M	A	DMS +PU+	M-DMS	n/a	n/a

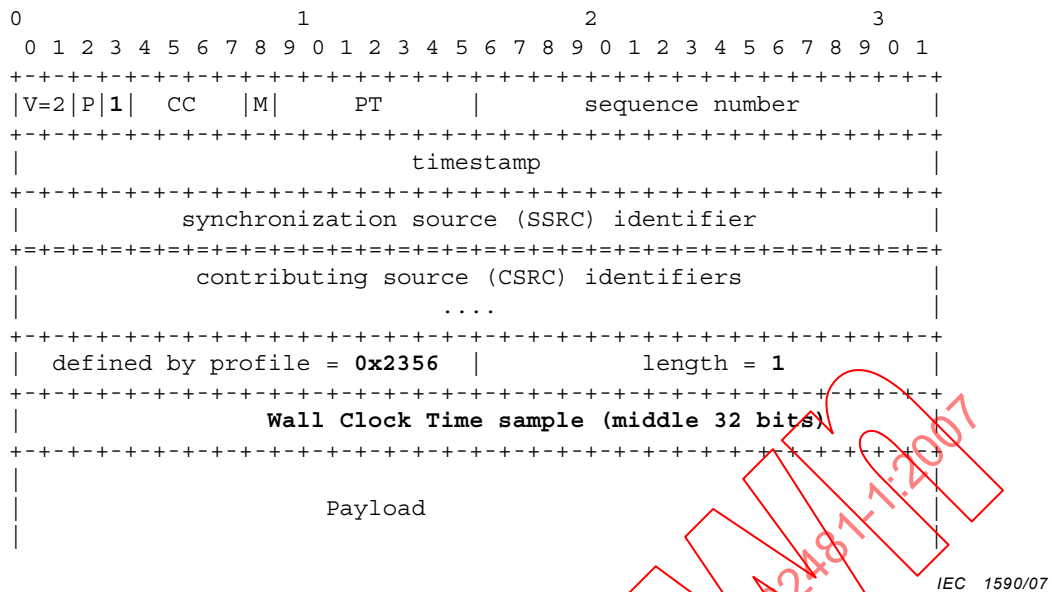


Figure 22 – Packet with wall clock time sample header extension

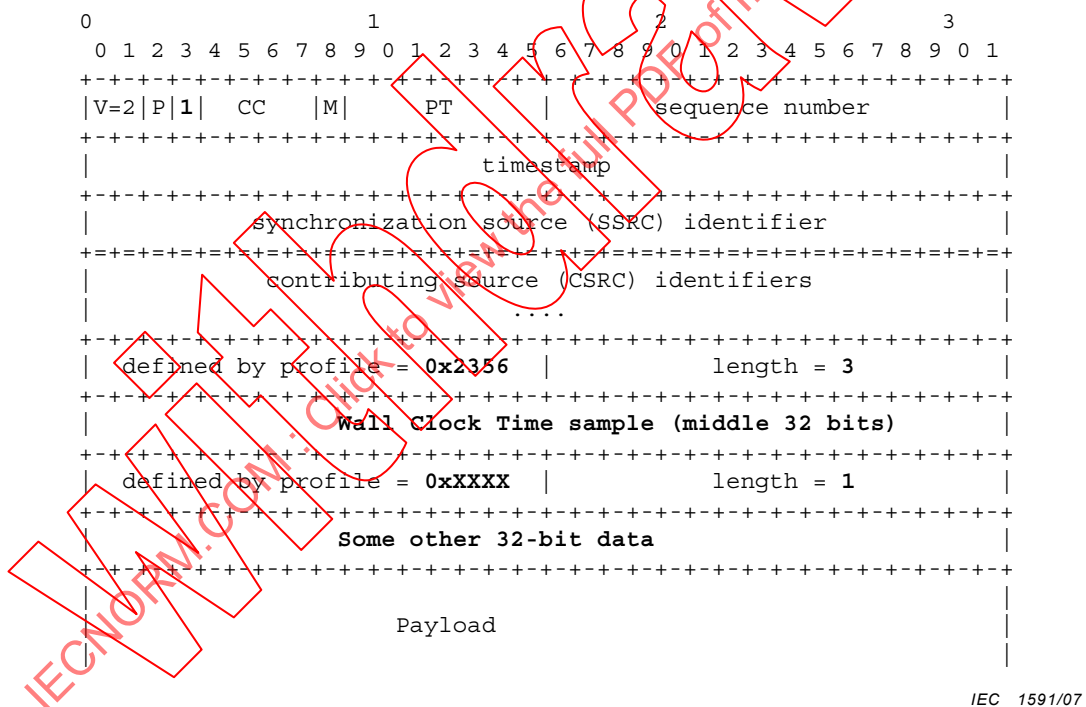


Figure 23 – Example of packet with another header extension following wall clock time sample

7.5.6.3.22 MT RTP pacing of RTP packets

7.5.6.3.22.1 In the absence of network congestion, the serving endpoint must be capable of delivering individual RTP packets to the network within 35 ms of a valid "target transmission time". "Target transmission time" is defined in 7.5.6.3.22.2.

The above does not apply when the serving endpoint performs transmission rate adaptation as described in 7.5.8.2.79.2 and 7.5.8.2.79.4.

The accuracy of the RTP packet delivery will be determined by capturing the observed network arrival time as defined in 7.5.6.3.18 and comparing it to a valid target transmission time.

M	A	DMS +PU+	M-DMS	n/a	n/a

The pace with which RTP packets are delivered to the network must be such that no pre-decoder buffer overflow can occur at the receiving endpoint.

This means that RTP packets must be paced in accordance with the standard decoder buffer model of the media format concerned. The standard decoder buffer model determines which are valid delivery times – "target transmission times" – for packets.

For TS/PS encapsulation a valid "target transmission time" is present in the RTP timestamp (see 7.5.7.4.7).

For ES encapsulation the "target transmission time" is not explicit in the packet, but may be derived from the multiplex of the source material (if present).

It should be noted that 35 ms is a hard limit, no packets are allowed to exceed these bounds.

7.5.6.3.22.2 For payload types other than the ones mentioned in 7.5.7.4.7, the "target transmission time" of an RTP packet must be defined as the intended delivery time of the first byte of its payload into the pre-decoder buffer of the receiving endpoint in a way that is consistent with the standard decoder buffer model applicable to the payload type.

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.6.3.23 MT RTP maximum reception packet rates

7.5.6.3.23.1 Upon reception of the header Max-Prate.dlna.org (see 7.5.8.2.28), the serving endpoint should understand this header and adjust (if necessary) the packet rate based on the max-packet-rate parameter, in order to conform to the maximum packet rate capability of the receiving endpoint.

S	A	DMS +PU+	M-DMS	n/a	n/a

This rule recommends the serving endpoint not to exceed the maximum packet rate specified by the receiving endpoint. Otherwise, the receiving endpoint may not be able to play all packets.

7.5.6.3.23.2 A serving endpoint that supports the Max-Prate.dlna.org header must also understand the feature tag dlna.Max-Prate (see 7.5.8.2.28.2) in a require header.

M	A	DMS +PU+	M-DMS	n/a	[33]

7.5.6.4 RTP receiving endpoint requirements

7.5.6.4.1 MT RTP interpret the P and X bits in the RTP header

Receiving endpoints must correctly interpret P bit and the X bit in the RTP packet header.

M	C	DMP DMR	M-DMP	MIU	[44]

If the P bit is 1, the receiving endpoint must remove the padding before processing the packet.

If the X bit is 1, the receiving endpoint must correctly parse the packet and process any headers that it understands, and tolerate any other headers.

7.5.6.4.2 MT RTP tolerate CSRC fields in the RTP header

Receiving endpoints must tolerate contributing sources (CSRCs) in the RTP packet header.

M	C	DMP DMR	M-DMP	MIU	[44]

Contributing sources are created only by RTP mixers, which are not required by DLNA.

7.5.6.4.3 MT RTP SSRC uniqueness – Receiving endpoints

Receiving endpoints must ignore SSRC collisions.

M	L	DMP DMR	M-DMP	MIU	[44]

RTP was designed for large multicast conferencing-type of applications.

SSRC collisions can occur in large conferences where multiple receivers/senders choose their own SSRC.

In a unicast environment, the RTP receiving endpoint must ignore SSRC collisions and continue the session with the negotiated SSRC values.

7.5.6.4.4 MT RTP expect random starting RTP sequence number

A receiving endpoint must accept an arbitrary starting sequence number.

M	C	DMP DMR	M-DMP	MIU	[44]

Subclause 5.1 of RFC 3550 [44] recommends random starting sequence numbers, so receiving endpoints must expect a random starting sequence number.

7.5.6.4.5 MT RTP expect random starting RTP timestamp

A receiving endpoint must accept an arbitrary starting RTP timestamp.

M	C	DMP DMR	M-DMP	MIU	[44]

Subclause 5.1 of RFC 3550 [44] recommends random starting RTP timestamps, so receiving endpoints must expect a random starting timestamp.

7.5.6.4.6 MT RTP required RTCP SDES items

The only RTCP SDES item required is CNAME.

M	C	DMP DMR	M-DMP	MIU	[44]

The CNAME item is required as per 6.1 of RFC 3550 [44].

7.5.6.4.7 MT RTP robust handling of RTCP BYE packet

A receiving endpoint must either accept or gracefully discard RTP data packets received after the reception of the RTCP BYE packet

M	C	DMP DMR	M-DMP	MIU	[44]

BYE packets may be received before or after the transmission of RTP packets has actually ended because of network reordering or retransmission.

7.5.6.4.8 MT RTP unknown RTP extensions

Receiving endpoints must tolerate RTP header extensions they do not support.

Tolerate means that the receiving endpoint is able to "parse and interpret" or "parse and ignore" header extensions.

M	R	DMP DMR	M-DMP	MIU	[44]

7.5.6.4.9 MT RTP out of order RTP packets and jitter conditions

7.5.6.4.9.1 Receiving endpoints receiving RTP packets must accept out-of-order packets.

"Must accept" means that the receiving endpoint is able to "receive and reorder" or "receive and ignore" header extensions.

M	R	DMP DMR	M-DMP	MIU	[44]

7.5.6.4.9.2 A receiving endpoint should be capable of processing RTP packets that arrive with up to 200 ms of total jitter.

S	C	DMP DMR	M-DMP	MIU	[44]

This jitter is the sum of the OS jitter (which is 35 ms) plus the network induced jitter.

7.5.6.4.9.3 If a receiving endpoint sends buffer fullness reports, it must tolerate variable rate transmission of RTP packets for a period up to the duration of the receiving endpoint's buffer.

M	A	DMP DMR	M-DMP	MIU	[44]

7.5.6.4.10 MT RTP/AVPF support

A receiving endpoint may use the RTP/AVPF profile, and send RTCP-based feedback messages of the types that serving endpoint has indicated are acceptable.

O	C	DMP DMR	M-DMP	MIU	[14]

The serving endpoint uses SDP to indicate if RTP/AVPF is used, and which feedback message types are acceptable.

7.5.6.4.11 MT RTP retransmission requests

If the RTP/AVPF profile is used, and if the serving endpoint has indicated that RTCP nack feedback messages are acceptable, then the receiving endpoint must use RTCP nack feedback messages to request any desired RTP packet retransmissions.

M	C	DMP DMR	M-DMP	MIU	[14]

7.5.6.4.12 MT RTP audio/rtx and video/rtx support

7.5.6.4.12.1 A receiving endpoint which requests retransmission of RTP packets should support the audio/rtx and video/rtx RTP payload formats for retransmitted packets.

S	A	DMP DMR	M-DMP	MIU	[16]

7.5.6.4.12.2 A serving endpoint that supports the audio/rtx and video/rtx RTP payload formats for retransmitted packets must use the SSRC-multiplexing method described in Clause 5 of [16] and must not use the session-multiplexing method (also described in Clause 5 of [16])

M	L	DMS +PU+	M-DMS	n/a	[16]

7.5.6.4.13 MT RTP packets that are retransmitted as identical copies

A receiving endpoint which requests retransmission of RTP packets may support that RTP packets are retransmitted as identical copies of the original RTP packets.

O	A	DMP DMR	M-DMP	MIU	n/a

7.5.6.4.14 MT RTP rules for counting RTP packets when retransmission requests are supported

7.5.6.4.14.1 A receiving endpoint which requests the retransmission of an RTP packet using an RTCP nack feedback message, must count that packet as a lost packet in RTCP Receiver Reports, even if the RTP packet is subsequently received.

M	A	DMP DMR	M-DMP	MIU	n/a

7.5.6.4.14.2 A receiving endpoint which requests the retransmission of an RTP packet using an RTCP nack feedback message, must not include that packet when computing the value for the "interarrival jitter" field in RTCP receiver reports, even if the RTP packet is subsequently received.

M	A	DMP DMR	M-DMP	MIU	n/a

7.5.6.4.15 MT RTP buffer fullness reports

7.5.6.4.15.1 If the RTP/AVPF profile is used, and if the serving endpoint has indicated that RTCP bfr feedback messages are acceptable, then the receiving endpoint should include buffer fullness reports (BFRs) with all RTCP receiver reports that it sends.

S	A	DMP DMR	M-DMP	MIU	[14]

BFRs can be used to detect network congestion and ensure the rendering endpoint's jitter buffer remains full.

7.5.6.4.15.2 A receiving endpoint that sends BFRs must send them at the rate that is in accordance with SDP provisions (if any).

M	A	DMP DMR	M-DMP	MIU	[47]

See 7.5.8.2.75.2 for SDP provisions that define the rate at which BFRs must be sent. The serving endpoint requires BFRs to be sent at this rate to detect congestion.

7.5.6.4.15.3 A receiving endpoint that sends BFRs must use the following syntax:

- an RTCP feedback message as defined in 6.1 of RTP/AVPF [14] must be used;
- the FMT field must be set to 3 indicating the bfr feedback message type;
- the PT field must be set to 205 indicating a transport layer feedback message;
- the feedback control information (FCI) section consists of 10 fields. The fields must occur in the FCI in the order listed below;
- N1: (32 bits) must be set to the number of free bytes in the receiving endpoint's network de-jitter buffer;
- N2: (32 bits) must be set to the current size, in bytes, of the receiving endpoint's network de-jitter buffer. The current size is the current maximum size and not the current utilization or amount of data in the buffer;
- N3: (16 bits) must be set to the amount of data queued in the receiving endpoint's network de-jitter buffer counted in milliseconds. If this information is not available, this field must be set to 0xFFFF. If the amount of data is more than 65534 ms, the fields must be set to 0xFFFFE;
- Playout delay: (16 bits) must be set to the difference between the scheduled playout time of the next ADU (application data unit) to be transferred to the coded data buffer or decoded if coded data buffer is not presented and the time of sending the BFR, as measured by the media playout clock, expressed in milliseconds. If this information is not available, the receiving endpoint must set this value to 0xFFFF. In case of an empty buffer, the playout delay is not defined and the field must be set to 0xFFFF;
- NSN: (16 bits) must be set to the RTP sequence number of the next ADU to be transferred or decoded for the SSRC reported on. In the case where the buffer does not contain any packets for this SSRC, the next not yet received RTP sequence number must be reported, i.e., an NSN value that is one larger than the least significant 16 bits of the RTCP SR or RR report block's "extended highest sequence number received";
- NUN: (16 bits) must be set to the unit number (within the RTP packet) of the next ADU to be transferred or decoded as defined in the payload format section of these guidelines, A/V media format profiles in the context of RTP transmission (7.5.7.2.1 to 7.5.7.2.6), for the given RTP payload format. The first unit in a packet has a unit number equal to zero. The unit number is incremented by one for each ADU in an RTP packet. In the case of RTP payload formats where each packet carries a single ADU, or where the ADU is not defined, the NUN field must be set to zero;

- D1: (32 bits) must be set to the number of free bytes in the receiving endpoint's coded data buffer, or 0 if the pre-decoder buffer is combined with the network de-jitter buffer. The field must be set to 0 if the RTSP Buffer-Info.dlna.org header did not provide a size for the coded data buffer;
- D2: (32 bits) must indicate the current size, in bytes, of the receiving endpoint's coded data buffer, or 0 if the pre-decoder buffer is combined with the network de-jitter buffer. The field must be set to 0 if the RTSP Buffer-Info.dlna.org header did not provide a size for the coded data buffer;
- D3: (16 bits) must be set to the amount of data queued in the receiving endpoint's coded data buffer counted in milliseconds. If this information is not available, this field must be set to 0xFFFF. If the amount of data is more than 65534 ms, the fields must be set to 0xFFFE;
- TD: (16 bits) must be set to the current value of target buffer duration, as defined in guideline 0. The field must be set to 0 if the Buffer-Info.dlna.org header was not included in the RTSP SETUP request, or if the "TD" parameter on that header was not present.

M	A	DMP DMR	M-DMP	MIU	[14]

The fields N2 and D2 are necessary because the receiving endpoint may decide to increase the sizes of these buffers as a result of headers in the PLAY response, or as a result of network congestion, for example.

See Figure 24 for more information.

The definition of application data unit is specific to each RTP payload format. See definitions in A/V media format profiles in the context of RTP transmission (7.5.7.2.1 to 7.5.7.2.6) and the comments about the NUN field, below.

In the two-buffer model where no RTP header is available in the coded data buffer, the reported NSN and the associated playout delay and NUN must use the next to be transferred RTP packet's sequence number in the transmission (network) jitter buffer.

For the NUN field:

For example in the case of H.264 (AVC), an ADU is defined as a NAL unit and for audio profiles, an ADU is an audio frame.

In the case of RTP payload formats where each packet carries a single ADU (for example H.263 and MPEG-4 visual simple profile) the NUN field will be set to zero.

MPEG-2 PS streams consist of a stream of bytes without payload-level framing. Therefore, NUN must always be 0 for such streams.

Future additions of media encoding or transports capable of having more than one ADU in each RTP payload must define what must be counted as an ADU for this format

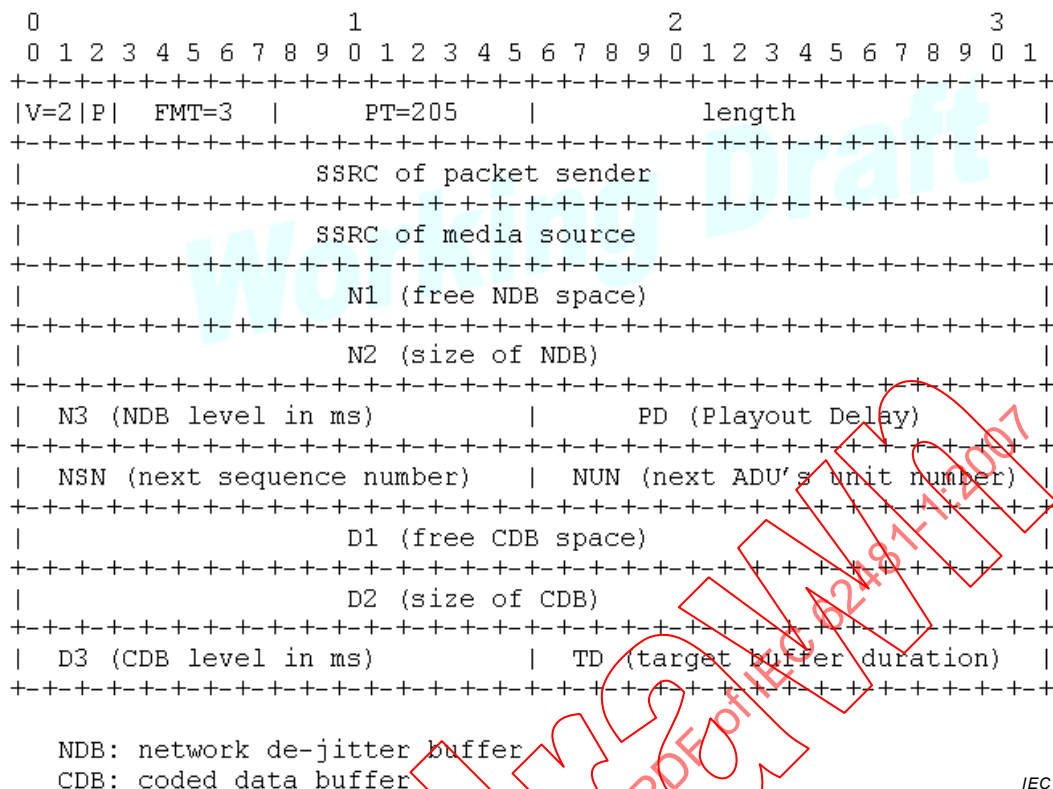


Figure 24 – BFR packet format

7.5.6.4.16 MT RTP tolerate wall clock time sample RTP header extension

The receiving endpoint must tolerate the wall clock time sample RTP header extension as defined in 7.5.6.3.15 through 7.5.6.3.21.

M	C	DMP	DMR	M-DMP	MIU	[44]

RFC 3550 [44] already mandates RTP header extensions must be silently ignored if they cannot be interpreted.

The aim of the referenced guidelines is for a receiving endpoint to actively use the wall clock time samples to perform clock recovery. As stated before, this is not mandatory.

7.5.7 RTP media transport – Adaptation of media format profiles

7.5.7.1 General

IEC 62481-2 specifies audio, video, and encapsulation characteristics for media content that conforms to the list of DLNA mandatory and optional media format profiles when the transport protocol is HTTP. A/V media format profiles in the context of RTP transmission (7.5.7.2.1 to 7.5.7.2.6) uses IEC 62481-2 as the basis and defines additional constraints and requirements necessary to reuse the profile ID values and exchange content using RTP. Readers should understand this clause as an adaptation layer or filtering layer to tailor the media format profiles defined in IEC 62481-2 for usage over RTP.

7.5.7.2 A/V media format profiles in the context of RTP transmission

7.5.7.2.1 MT RTP profile ID usage

For transport over RTP, A/V content must use the same profile ID values defined in IEC 62481-2.

M	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

7.5.7.2.2 MT RTP MPEG-2 media format profiles

7.5.7.2.2.1 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- MPEG-2 video with any type of companion audio;
- Full single program transport stream encapsulation;
- DLNA transport packets without timestamp fields (188 byte ISO profiles).

The RTP encapsulation must use either

- the payload format MP2T as defined in [31] and [46] with constraints in 7.5.7.4.2; or
- the payload format vnd.dlna.mpeg-mp2t as defined in 7.5.7.4.3.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2, [31]

Full single program transport streams with zero TTS carrying MPEG-2 media format profiles.

7.5.7.2.2.2 For A/V profile ID values in IEC 62481-2 for which the following holds:

- MPEG-2 video with any type of companion audio;
- partial single program transport stream encapsulation;
- DLNA transport packets without timestamp fields (188-byte ISO profiles)

The RTP encapsulation must use payload format vnd.dlna.mpeg-mp2t as defined in 7.5.7.4.3.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

Partial single program transport streams with zero TTS carrying MPEG-2 media format profiles.

7.5.7.2.2.3 For A/V profile ID values in IEC 62481-2 for which the following holds:

- MPEG-2 video with any type of companion audio;
- partial or full single program transport stream encapsulation;
- non-zero TTS.

The RTP encapsulation must use payload format vnd.dlna.mpeg-tts as defined in 7.5.7.4.3.

MT RTP vnd.dlna.mpeg-tts RTP payload format .

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

Partial or full single program transport streams with non-zero TTS carrying MPEG-2 media format profiles.

7.5.7.2.2.4 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- MPEG-2 video with any type of companion audio;
- program stream encapsulation.

The RTP encapsulation must use payload format MP2P as defined in [31] and [46].

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2, [31] [46]

Program streams carrying MPEG-2 media format profiles.

7.5.7.2.2.5 For A/V profile ID values in IEC 62481-2 for which the following holds:

- MPEG-2 video with any type of companion audio;
- MPEG-2 elementary stream encapsulation.

The RTP encapsulation for video must use payload format MPV as defined in [31] and the RTP encapsulation for audio must comply with Adaptation of audio-only media format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams (7.5.7.3.1 to 7.5.7.3.11).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2, [31]

MPEG-2 elementary streams carrying MPEG-2 media format profiles.

7.5.7.2.3 MT RTP AVC media format profiles

7.5.7.2.3.1 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- AVC video with any type of companion audio;
- full single program transport stream encapsulation;
- DLNA transport packets without timestamp fields (188-byte ISO profiles).

The RTP encapsulation must use either

- the payload format MP2T as defined in [[31] and [46]] with constraints in 7.5.7.4.2; or
- the payload format vnd.dlna.mpeg-mp2t as defined in 7.5.7.4.3.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2, [31]

Full single program transport streams with zero TTS carrying AVC media format profiles.

7.5.7.2.3.2 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- AVC video with any type of companion audio;
- partial single program transport stream encapsulation;
- DLNA transport packets without timestamp fields (188-byte ISO profiles).

The RTP encapsulation must use payload format vnd.dlna.mpeg-mp2t as defined in 7.5.7.4.3 with the following additional constraints. The TS packet size must be 188 bytes and the 4-byte TTS defined in IEC 62481-2 must be excluded.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

Partial single program transport streams with zero TTS carrying AVC media format profiles.

7.5.7.2.3.3 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- AVC video with any type of companion audio;
- partial or full single program transport stream encapsulation;

- non-zero TTS.

The RTP encapsulation must use payload format vnd.dlna.mpeg-tts as defined in 7.5.7.4.4.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

Partial or full single program transport streams with non-zero TTS carrying AVC media format profiles.

7.5.7.2.3.4 For A/V profile ID values in IEC 62481-2 for which the following holds:

- AVC video with any type of companion audio;
- profile ID indicating MP4 or 3GPP encapsulation.

When these profile ID values are used in conjunction with RTP all of the following must apply.

- Audio and video must be encapsulated as elementary streams (no use of the MPEG-4 file format or 3GPP file format)
- The RTP encapsulation for video must use payload format H264 [49] with constraints in Guidelines for the encapsulation for AVC elementary streams (7.5.7.6.1 to 7.5.7.6.8).
- The RTP encapsulation for audio must comply with Adaptation of audio-only media format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams (7.5.7.3.1 to 7.5.7.3.11).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2 [49]

AVC media format profiles that indicate MP4 or 3GPP encapsulation

7.5.7.2.4 MT RTP MPEG-4 Part 2 media format profiles

7.5.7.2.4.1 For A/V profile ID values in IEC 62481-2 for which the following holds:

- MPEG-4 Part 2 video with any type of companion audio;
- full single program transport stream encapsulation;
- DLNA transport packets without timestamp fields (188 byte ISO profiles).

The RTP encapsulation must use either

- the payload format MP2T [[31], [46]] with constraints in 7.5.7.4.2; or
- the payload format vnd.dlna.mpeg-mp2t as defined in 7.5.7.4.3.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2, [31]

Full single program transport streams with zero TTS carrying MPEG-4 Part 2 media format profiles.

7.5.7.2.4.2 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- MPEG-4 Part 2 video with any type of companion audio;
- partial single program transport stream encapsulation;
- DLNA transport packets without timestamp fields (188-byte ISO profiles).

The RTP encapsulation must use payload format vnd.dlna.mpeg-mp2t as defined in 7.5.7.4.3.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

Partial single program transport streams with zero TTS carrying MPEG-4 Part 2 media format profiles.

7.5.7.2.4.3 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- MPEG-4 Part 2 video with any type of companion audio;
- partial or full single program transport stream encapsulation;
- non-zero TTS.

The RTP encapsulation must use payload format vnd.dlna.mpeg-tts as defined in 7.5.7.4.4 with the following additional constraints. The TS packet size must be 192 bytes and the 4-byte TTS defined in IEC 62481-2 must be included.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

Partial or full single program transport streams with non-zero TTS carrying MPEG-4 Part 2 media format profiles.

7.5.7.2.4.4 For A/V Profile ID values in IEC 62481-2 for which the following holds:

- MPEG-4 Part 2 video with any type of companion audio;
- profile ID indicating MP4 or ASF encapsulation.

When these profile ID values are used in conjunction with RTP all of the following must apply:

- audio and video must be encapsulated as elementary streams (no use of the MPEG-4 or ASF file format);
- the RTP encapsulation for video must use payload format mpeg4-generic [48] with constraints in Guidelines for the encapsulation for MPEG-4 part 2 elementary streams (7.5.7.7.1 to 7.5.7.7.5);
- the RTP encapsulation for audio must comply with Adaptation of audio-only media format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams (7.5.7.3.1 to 7.5.7.3.11).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2, [48]

MPEG-4 Part 2 media format profiles that indicate MP4 or ASF encapsulation.

7.5.7.2.5 MT RTP H.263 media format profiles

For A/V profile ID values in IEC 62481-2 for which the following holds:

- H.263 video with any type of companion audio;
- profile ID indicating MP4 encapsulation.

When these profile ID values are used in conjunction with RTP all of the following must apply:

- audio and video must be encapsulated as elementary streams (no use of the MPEG-4 file format);
- the RTP encapsulation for video must use payload format H263-2000 [46], [36] with constraints in Guidelines for the encapsulation of H.263 streams (7.5.7.9.1 to 7.5.7.9.4);

- the RTP encapsulation for audio must comply with Adaptation of audio-only media format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams (7.5.7.3.1 to 7.5.7.3.11).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2, [36] [46]

H.263 media format profiles that indicate MP4 encapsulation

7.5.7.2.6 MT RTP WMV media format profiles

For A/V profile ID values in IEC 62481-2 for which the following holds:

- WMV video with any type of companion audio;
- media format profiles indicating use of ASF encapsulation.

When these profile ID values are used in conjunction with RTP all of the following must apply:

- audio and video must be encapsulated as elementary streams (no use of the ASF file format);
- the RTP encapsulation for video must use payload format WMV [66] with constraints in the Guidelines for encapsulation of WMA and WMV elementary streams (7.5.7.5.1 to 7.5.7.5.7),
- the RTP encapsulation for audio must comply with Adaptation of audio-only media format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams (7.5.7.3.1 to 7.5.7.3.11).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2 [66]

WMV media format profiles.

7.5.7.3 Adaptation of audio-only media format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams.

7.5.7.3.1 MT RTP media format profile ID usage

For transport over RTP, audio-only content must use the same profile ID values defined in IEC 62481-2.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

7.5.7.3.2 MT RTP audio elementary streams of A/V media format profiles

Audio components of A/V profiles in IEC 62481-2 transmitted over RTP as separate elementary streams must follow the adaptation guidelines defined in Adaptation of audio-only media format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams (7.5.7.3.1 to 7.5.7.3.11).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

7.5.7.3.3 MT RTP elementary streams of audio-only media format profiles

Audio content of audio-only profiles in IEC 62481-2 must be transmitted over RTP as elementary streams using the adaptation guidelines defined in Adaptation of audio-only media

format profiles to RTP, and adaptation of the audio component of A/V media format profiles when transmitted as separate elementary streams (7.5.7.3.1 to 7.5.7.3.11).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2

7.5.7.3.4 MT RTP LPCM media format profiles

The RTP encapsulation for LPCM must use payload format L16 [45].

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[45]

7.5.7.3.5 MT RTP MP3,MP3X, MPEG-2 L2 media format profiles

The RTP encapsulation for MP3, MP3X, and MPEG-2 L2 must use payload format MPA [31].

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[31]

7.5.7.3.6 MT RTP AC3, XAC3 media format profiles

The RTP encapsulation for AC-3 and XAC3 must use payload format ac3 [15].

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[67] [15]

7.5.7.3.7 MT RTP AAC, HE-AAC media format profiles

The RTP encapsulation for AAC and HE-AAC must use payload format mpeg4-generic [48] with constraints in Guidelines for the encapsulation of MPEG-4 AAC streams (7.5.7.8.1 to 7.5.7.8.10).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[48]

7.5.7.3.8 MT RTP G726 media format profiles

The RTP encapsulation for G726 must use payload format G726-32 [45].

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[45]

7.5.7.3.9 MT RTP WMA media format profiles

The RTP encapsulation for WMA must use payload format WMA [66] with constraints in Guidelines for encapsulation of WMA and WMV elementary streams (7.5.7.5.1 to 7.5.7.5.7).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[66]

7.5.7.3.10 MT RTP AMR media format profiles

The RTP encapsulation for AMR must use payload format AMR [42] with constraints in Guidelines for the encapsulation AMR streams (7.5.7.10.1 to 7.5.7.10.4).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2 [42]

7.5.7.3.11 MT RTP AMR- WBPlus media format profiles

The RTP encapsulation for AMR-WBPlus must use payload format AMR-WB+ [37] with constraints in Guidelines for the encapsulation AMR-WBplus streams (7.5.7.11.1 to 7.5.7.11.6).

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	IEC 62481-2 [37]

7.5.7.4 Guidelines for encapsulation of MPEG-2 transport streams

7.5.7.4.1 MT RTP payload formats for MPEG TS encapsulated content

7.5.7.4.1.1 The guidelines in 7.5.7.4.1 apply to 7.5.7.4.2, 7.5.7.4.3 and 7.5.7.4.4

M	L	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1 [31]

7.5.7.4.1.2 All RTP Timestamps for MPEG TS encapsulated content will be in 90 kHz clock units.

M	C	DMS +PU+	M-DMS	n/a	[31]

7.5.7.4.1.3 The 90 kHz clock utilized as the basis for the RTP header timestamp field must be synchronized with the MPEG system clock of the content.

M	L	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1 [31]

7.5.7.4.1.4 The RTP timestamp values may or may not be equal to the MPEG program_clock_reference_base value in the PCR field.

O	C	DMS +PU+	M-DMS	n/a	[31]

7.5.7.4.1.5 An ADU (application data unit) must be one complete TS packet as specified in the guidelines for the RTP payload format in use.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[31]

The size of a TS packet differs depending on the RTP payload format in use.

7.5.7.4.2 MT RTP MP2T RTP payload format definition

7.5.7.4.2.1 The MP2T RTP payload format can only be used to transfer full SPTS streams in RTP. If this RTP payload format definition is used, the RTP packets must meet the following guidelines.

M	L	DMS +PU+	M-DMS	n/a	[31]

This RTP payload format definition is a strict definition of RFC2250 using full SPTS streams.

7.5.7.4.2.2 Each MPEG TS packet must be 188 bytes in length and conform to the requirements of ISO/IEC 13818-1. The MP2T payload format can only be utilized with DLNA media format profiles that utilize MPEG2 TS encapsulation and have 188 byte TS packets. Specifically MP2T is not a valid RTP payload format for DLNA media format profiles utilizing the 192-byte TS packet syntax.

M	C	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1

7.5.7.4.2.3 The RTP timestamp must represent the ideal transmission time of the first byte of the first MPEG TS packet contained within the RTP payload as defined in RFC2250 and further clarified below.

M	C	DMS +PU+	M-DMS	n/a	[31]

7.5.7.4.2.4 The ideal transmission time of the first byte of an MPEG2 TS Packet must be calculated as follows:

$$TSTimestampY = \text{Offset} + 90\text{KHZX} + ((\Delta\text{TPCR} / N) * I) \quad (2)$$

where

I is the index value of the TS packet relative to the last TS packet containing a PCR value. The first TS packet after a PCR value will have an index of 1, the second TS packet an Index of 2, and so on;

N is the number of TS packets between TS packets with PCR values plus one for the TS packet with the PCR value. For example, if a SPTS has 2 000 TS packets per second, and PCR values occur every 100 ms, the value of N would be 200;

90KHZX is the value of the 90 KHz clock associated with the last TS packet containing a PCR value;

ΔTPCR is the delta time value between PCR values as defined in Clause 3;

Offset is the RTP timestamp offset as defined in 7.5.6.3.1.

M	C	DMS +PU+	M-DMS	n/a	n/a

7.5.7.4.2.5 The serving endpoint must utilize either a static payload ID of '33' or a dynamic payload format of 'MP2T' in the SDP media description fields as defined in 7.5.8.2.66.

M	C	DMS +PU+	M-DMS	n/a	n/a

7.5.7.4.3 MT RTP vnd.dlna.mpeg-mp2t RTP payload format definition

7.5.7.4.3.1 The vnd.dlna.mpeg-mp2t RTP payload format definition can be used to transfer partial or full SPTS streams in RTP. If this TS payload definition is used, the RTP packets must meet the following guidelines 7.5.7.4.3.2, 7.5.7.4.3.3 and 7.5.7.4.3.4.

M	L	DMS +PU+	M-DMS	n/a	n/a

This RTP TS payload format definition is a looser definition of RFC2250 MP2T that allows both partial and full SPTS streams. The RTP timestamp calculations are identical even though the temporal relationship between any two adjacent TS packets is not always the same.

7.5.7.4.3.2 This payload format definition is identical to MP2T as defined in 7.5.7.4.2.2, except for the name and that it can also be used to carry partial SPTS streams.

M	L	DMS +PU+	M-DMS	n/a	n/a

7.5.7.4.3.3 The RTP packet timestamp must be calculated in the same manner as described in 7.5.7.4.2.3 and 7.5.7.4.2.4 for the MP2T payload format definition.

M	L	DMS +PU+	M-DMS	n/a	n/a

7.5.7.4.3.4 The serving endpoint must utilize a dynamic payload format of 'vnd.dlna.mpeg-mp2t' in the SDP media description fields as defined in 7.5.8.2.66.

M	L	DMS +PU+	M-DMS	n/a	n/a

7.5.7.4.3.5 To reduce jitter on the transmission of PCR timestamps, the serving endpoint should send RTP packets as soon as it contains a TS packet with a PCR, without waiting for the max payload size to be reached.

S	A	DMS +PU+	M-DMS	n/a	n/a

7.5.7.4.4 MT RTP vnd.dlna.mpeg-tts RTP payload format definition

7.5.7.4.4.1 The vnd.dlna.mpeg-tts RTP payload format definition can be used to transfer partial or full SPTS streams in RTP. If this RTP payload format definition is used, the RTP packets must meet guidelines 7.5.7.4.4.2, 7.5.7.4.4.7 and 7.5.7.4.4.8.

M	A	DMS +PU+	M-DMS	n/a	n/a

This RTP payload format definition is designed to preserve individual TS packet timing references in a partial SPTS stream.

7.5.7.4.4.2 The vnd.dlna.mpeg-tts RTP payload format definition can only be utilized with DLNA media format profiles that utilize MPEG2 TS encapsulation and have 192 byte TS packets. It must be used for DLNA media format profiles utilizing the DLNA-defined 192 byte TS packet format with timestamp.

M	A	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1

The individual timestamps in each TS packet are designed to preserve the temporal relationship between each TS packet and its two adjacent packets.

7.5.7.4.4.3 Each TS packet must be 192 bytes in length consisting of a 4 byte TTS timestamp field followed by a 188 byte MPEG TS packet conforming to the requirements of ISO/IEC 13818-1.

M	A	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1

7.5.7.4.4.4 The TTS timestamp field must be in 27 MHz clock units. The 27 MHz clock utilized as the basis for the TTS timestamps must be synchronized with the MPEG system clock of the content.

M	A	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1

7.5.7.4.4.5 The TTS timestamp value is a 32-bit binary counter and is not defined using the same syntax as the PCR field. If this TS definition is utilized, the serving endpoint must ensure that the individual TTS timestamp values have an accuracy of ± 500 ns.

M	A	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1

7.5.7.4.4.6 The TTS timestamp field must contain valid timestamp values.

M	A	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1

It should be noted that this is different from how the TTS timestamps are used in HTTP.

7.5.7.4.4.7 The RTP timestamp must be derived from the TTS timestamp of the first TS packet in the RTP payload using the following equations:

For the first RTP timestamp (TSRTP) in the RTP stream, or after a program system clock discontinuity:

$$\text{TSRTP} = \text{Offset} + (\text{TSTTS}/300)$$

where offset is the RTP offset defined in 7.5.7.4.1 and TSTTS is the 27 MHz TTS 32-bit timestamp.

For subsequent RTP timestamps:

$$\text{TSRTP}_N = \text{TSRTP}_{N-1} + ((\text{TSTTS}_N - \text{TSTTS}_{N-1}) / 300)$$

where

TSRTP_N is the RTP timestamp of the Nth RTP packet;

TSTTS_N is the TTS timestamp of the first TS packet in the payload of the Nth RTP packet.

M	A	DMS +PU+	M-DMS	n/a	n/a

These equations compensate for the difference in time base between the TTS timestamp (27 MHz) and the RTP timestamp (90 kHz). It allows the RTP timestamp to reach its full value (roll over at 232). Implementers must account for roller in the $\text{TSTTS}_N - \text{TSTTS}_{N-1}$ expression.

7.5.7.4.4.8 The serving endpoint must utilize a dynamic payload format of 'vnd.dlna.mpeg-tts' in the SDP media description fields as defined in 7.5.8.2.66.

M	L	DMS +PU+	M-DMS	n/a	n/a

7.5.7.4.5 MT RTP payload format support for TS without TTS

If a receiving endpoint claims to support TS without TTS then it must support both RTP payload format definitions MP2T and vnd.dlna.mpeg-mp2t.

M	A	DMP DMR	M-DMP	MIU	n/a

For full TS its up to the serving endpoint whether MP2T or vnd.dlna.mpeg-mp2t is used, this implies the receiving endpoint must be able to accept either one.

7.5.7.4.6 MT RTP clock accuracy for MPEG-2 TS and MPEG-2 PS

The RTP timestamps applied to MPEG-2 TS and PS encapsulated content must utilize a 90 KHz clock source that is accurate to within ± 30 parts per million (ppm) and have a slew rate of less than 0,075 Hz per second as defined in ISO/IEC 13818-1.

M	C	DMS +PU+	M-DMS	n/a	ISO/IEC 13818-1 [31]

Only that way the clock recovered by the receiving endpoint can meet the specifications needed for decoding.

7.5.7.4.7 MT RTP target transmission time for MPEG TS and PS encapsulated content

For payload types MP2P, MP2T, vnd.dlna.mpeg-mp2t, and vnd.dlna.mpeg-tts, the "target transmission time" of an RTP packet is defined by its RTP timestamp field.

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.7.5 Guidelines for encapsulation of WMA and WMV elementary streams

7.5.7.5.1 MT RTP ADU usage

An ADU must be one complete MAU (media access unit) as specified in [66].

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[66]

7.5.7.5.2 MT RTP WMA time stamps in ASF are presentation time stamps

7.5.7.5.2.1 The "presentation time" field of a WMA media object in ASF must be treated as a presentation time. The "timestamp" field in the RTP header must be set to a value reflecting the presentation time of the first payload in the RTP packet.

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[65] [66]

7.5.7.5.2.2 The "decode time" field must not be included in the RTP payload format header, unless the decode time of the WMA media object is known.

M	L	DMS +PU+	M-DMS	n/a	[65] [66]

When WMA is stored in an ASF container, decode times of WMA media objects are not available.

7.5.7.5.3 MT RTP WMV time stamps in ASF are decode time stamps

7.5.7.5.3.1 The "presentation time" field of a WMV media object in ASF must be treated as a decode time. The "timestamp" field in the RTP header must be set to a value reflecting the decode time of the first payload in the RTP packet.

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[65] [66]

7.5.7.5.3.2 The a=fmtp line in SDP must specify ts=DTS to indicate this usage.

M	C	DMS +PU+	M-DMS	n/a	[66]

7.5.7.5.3.3 The "presentation time" field of a WMV media object in ASF must not be used as the presentation time field of the RTP packet.

M	L	DMS +PU+	M-DMS	n/a	[65] [66]

When WMV is stored in an ASF container, presentation times of WMV media objects are not explicitly available.

7.5.7.5.4 MT RTP determining the value of the SDP "profile" parameter for WMV

If the ASF meta data object exists, and if it contains a "DeviceConformanceTemplate" attribute for the WMV media stream, then that attribute may be used to determine the value for the "profile" parameter in the WMV SDP syntax.

The first two characters in the "DeviceConformanceTemplate" value determine the SDP "profile" parameter as follows:

- SP: profile=0
- MP: profile=1

O	C	DMS +PU+	M-DMS	n/a	[65] [66]

7.5.7.5.5 MT RTP determining the value of the SDP "level" parameter for WMV

If the ASF meta data object exists, and if it contains a "DeviceConformanceTemplate" attribute for the WMV media stream, then that attribute may be used to determine the value for the "level" parameter in the WMV SDP syntax.

The last two characters in the "DeviceConformanceTemplate" value determine the SDP "level" parameter as follows:

- LL: level=0;
- ML: level=1;
- HL: level=2.

O	C	DMS +PU+	M-DMS	n/a	[65] [66]

7.5.7.5.6 MT RTP determining the value of the SDP "aspect" parameter for WMV

If the ASF meta data object exists, and if it contains both an "AspectRatioX" attribute and an "AspectRatioY" attribute for the WMV media stream, then those attributes can be used to determine the value for the "aspect" parameter in the WMV SDP syntax.

O	C	DMS +PU+	M-DMS	n/a	[65] [66]

7.5.7.5.7 MT RTP determining the value of the SDP "profile" parameter for WMA

If the ASF meta data object exists, and if it contains a "DeviceConformanceTemplate" attribute for the WMA media stream, then that attribute may be used to determine the value for the "profile" parameter in the WMA SDP syntax.

If the first character in the "DeviceConformanceTemplate" value is "L", then the second character may be used as the value for the SDP "profile" parameter.

O	C	DMS +PU+	M-DMS	n/a	[65] [66]

7.5.7.6 Guidelines for the encapsulation for AVC elementary streams

7.5.7.6.1 MT RTP H.264 RTP payload format

If H.264/AVC elementary streams are transmitted over RTP, then RFC 3984 must be used as specified in 7.5.7.6.2 to 7.5.7.6.8 inclusive.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[49]

7.5.7.6.2 MT RTP H.264 ADU usage

An ADU must be a complete a NAL unit.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[49]

7.5.7.6.3 MT RTP H.264 packetization

A serving endpoint must use one of the following packetization modes:

- single NAL unit mode;
- non-interleaved mode;
- interleaved mode.

M	C	DMS +PU+	M-DMS	n/a	[49]

For the single NAL unit mode, non-interleaved mode, and interleaved mode, the values of the packetization-mode MIME/SDP parameter are equal to 0, 1, and 2, respectively.

7.5.7.6.4 MT RTP H.264 interleaved mode

7.5.7.6.4.1 If rtp-h264-deint-buf-cap.dlna.org header (see 7.5.7.6.7) is not present in the RTSP DESCRIBE request, then a serving endpoint must set the value of sprop-interleaving-depth parameter in the a=fmtp line of the SDP equal to 0.

M	L	DMS +PU+	M-DMS	n/a	[49]

When sprop-interleaving-depth equals to 0, the interleaved packetization mode can be used to encapsulate coded data from multiple coded pictures into the same RTP payload, without incurring any implementation complexity associated with interleaving of data. For low bit-rate media streams, this aggregation mechanism may help in avoiding a drop in the available bit rate for the whole WLAN segment and decreases the RTP/UDP/IP header overhead for the RTP stream.

7.5.7.6.4.2 A receiving endpoint must support the interleaved packetization mode with the value of the sprop-interleaving-depth parameter in the a=fmtp line of the SDP equal to 0.

M	A	DMP DMR	M-DMP	MIU	[49]

7.5.7.6.5 MT RTP H.264 single NAL unit mode

A receiving endpoint must support the single NAL unit packetization mode.

M	C	DMP DMR	M-DMP	MIU	[49]

7.5.7.6.6 MT RTP H.264 non-interleaved mode

A receiving endpoint must support the non-interleaved packetization mode.

M	A	DMP DMR	M-DMP	MIU	[49]

7.5.7.6.7 MT RTP H.264 de-interleaving buffer capability

If a receiving endpoint supports the interleaved packetization mode and a value of the sprop-interleaving-depth MIME/SDP parameter greater than 0, then the receiving endpoint must include the rtp-h264-deint-buf-cap.dlna.org header in the DESCRIBE request. The syntax of the header is specified as follows:

- rtp-h264-deint-buf-cap.dlna.org = "rtp-h264-deint-buf-cap.dlna.org" *LWS ":" *LWS 1*DIGIT

The value of rtp-h264-deint-buf-cap.dlna.org must be in the range of 0 to 4294967295, inclusive.

The value of the rtp-h264-deint-buf-cap.dlna.org header is interpreted identically to the deint-buf-cap MIME/SDP parameter specified in RFC 3984.

Example:

- rtp-h264-deint-buf-cap.dlna.org: 32768

This specifies that the receiving endpoint is capable of supporting such RTP streams that require a de-interleaving buffer up to 32768 bytes (inclusive).

M	A	DMP DMR	M-DMP	MIU	[49]

This indicates how big a buffer a receiving endpoint can allocate for the de-interleaving process of interleaved H.264 RTP streams.

7.5.7.6.8 MT RTP H.264 de-interleaving buffer requirement

A serving endpoint must set the value of the sprop-deint-buf-req parameter of H.264/AVC in the a=fmtp line of the SDP equal to or less than the value of the rtp-h264-deint-buf-cap.dlna.org header in the RTSP DESCRIBE request.

M	L	DMS +PU+	M-DMS	n/a	[49]

7.5.7.7 Guidelines for the encapsulation for MPEG-4 part 2 elementary streams

7.5.7.7.1 MT RTP payload format for MPEG-4 Part 2 elementary streams

If MPEG-4 Part 2 elementary streams are transmitted over RTP, then the RTP payload format mpeg4-generic must be used as specified in guidelines 7.5.7.7.2 to 0, inclusive.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[48]

"mpeg4-generic" is defined in RFC 3640.

7.5.7.7.2 MT RTP MPEG-4 Part 2 uses generic mode

A serving endpoint that transmits MPEG-4 Part 2 elementary streams over RTP must only use the "generic" mode of the RTP payload format mpeg4-generic.

M	C	DMS +PU+	M-DMS	n/a	[48]

Generic mode is the only mode defined in RFC 3640 that is applicable to MPEG-4 Part 2.

The usage of generic mode is signaled by "mode=generic" on the a=fmtp line in SDP. See 3.3.2 of RFC 3640.

7.5.7.7.3 MT RTP MPEG-4 Part 2 ADU usage

An ADU must be a complete AU (access unit) as specified in RFC 3640.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[48]

7.5.7.7.4 MT RTP MPEG-4 Part 2 concatenation and fragmentation of access units

A serving endpoint may concatenate and fragment MPEG-4 access units as defined in 2.3 and 2.4 of RFC 3640 [48].

O	C	DMS +PU+	M-DMS	n/a	[48]

7.5.7.7.5 MT RTP MPEG-4 Part 2 interleaving of access units

A serving endpoint must not interleave MPEG-4 access units as defined in 2.5 of RFC 3640. This means that if the "AU-Index-delta" field is present in the AU header, then its value must be set to 0. Additionally, if the SDP parameter maxDisplacement is present on the a=fmtp line, then its value must be 0.

M	L	DMS +PU+	M-DMS	n/a	[48]

Setting "AU-Index-delta" to 0 means that the index number of the access unit is equal to the previous index number plus one.

7.5.7.8 Guidelines for the encapsulation of MPEG-4 AAC streams

7.5.7.8.1 MT RTP payload format for MPEG-4 AAC LC and LTP streams

If MPEG-4 AAC LC or LTP streams are transmitted over RTP, then the RTP payload format RFC 3640 must be used as specified in 7.5.7.8.2 to 7.5.7.8.10 inclusive.

M	A	DMS +PU+	M-DMS	n/a	[48]

7.5.7.8.2 MT RTP MPEG-4 AAC

A serving endpoint that transmits MPEG-4 AAC streams over RTP must use the high bit-rate AAC mode of the RTP payload format RFC 3640.

M	L	DMS +PU+	M-DMS	n/a	[48]

The usage of high bit-rate AAC mode is signalled by "mode=AAC-hbr" on the a=fmtp line in SDP. See 3.3.5 of RFC 3640.

Alternative, low bit-rate AAC mode could have been used only when AAC frame length is at most 63 octets, i.e., corresponding to such low bit rates that are used rarely in DLNA. Therefore, low bit-rate mode is not allowed.

7.5.7.8.3 MT RTP MPEG-4 AAC concatenation and fragmentation of access units

7.5.7.8.3.1 A serving endpoint may concatenate and fragment MPEG-4 AAC access units as defined in 2.4 of RFC 3640.

O	R	DMS +PU+	M-DMS	n/a	[48]

7.5.7.8.3.2 If a serving endpoint concatenates or fragments MPEG-4 AAC access units, RTP packets must not contain fragments of multiple access units.

M	R	DMS +PU+	M-DMS	n/a	[48]

7.5.7.8.4 MT RTP MPEG-4 AAC non-interleaving of access units

A serving endpoint must support non-interleaved packetization mode.

M	C	DMS +PU+	M-DMS	n/a	[48]

7.5.7.8.5 MT RTP MPEG-4 AAC interleaving of access units

If a serving endpoint supports interleaving then it must apply interleaving when rtp-aac-deint-buf-cap.dlna.org header is present in the RTSP DESCRIBE request.

M	A	DMS +PU+	M-DMS	n/a	[48]

7.5.7.8.6 MT RTP MPEG-4 AAC RTP interleaving in SDP

A serving endpoint must set the value of the maxDisplacement parameter in the a=fmtp line of the SDP equal to or less than the value of the rtp-aac-deint-buf-cap.dlna.org header in the RTSP DESCRIBE request.

M	A	DMS +PU+	M-DMS	n/a	[48]

The value of the SDP maxDisplacement parameter in a=fmtp can be 0.

7.5.7.8.7 MT RTP MPEG-4 AAC non-interleaving of access units

When a serving endpoint uses non-interleaved mode then if the "AU-Index-delta" field is present in the AU header, then its value must be 0. Additionally, if the SDP parameter maxDisplacement is present on the a=fmtp line, then its value must be 0.

M	C	DMS +PU+	M-DMS	n/a	[48]

Setting "AU-Index-delta" to 0 means that the index number of the access unit is equal to the previous index number plus one.

7.5.7.8.8 MT RTP MPEG-4 AAC non-interleaved mode

A receiving endpoint must support the non-interleaved packetization mode.

M	A	DMP DMR	M-DMP	MIU	[48]

7.5.7.8.9 MT RTP MPEG-4 AAC interleaved mode

A receiving endpoint may support the interleaved packetization mode.

O	A	DMP DMR	M-DMP	MIU	[48]

7.5.7.8.10 MT RTP MPEG-4 AAC RTP interleaving indication

If a receiving endpoint supports interleaving then it must include the rtp-aac-deint-buf-cap.dlna.org header in the DESCRIBE request.

The syntax of this header is specified as follows:

- rtp-aac-deint-buf-cap-line = "rtp-aac-deint-buf-cap.dlna.org" *LWS ":" *LWS 1*DIGIT

The value of the rtp-aac-deint-buf-cap.dlna.org header is interpreted identically to the maxDisplacement SDP parameter, specified in [48].

M	A	DMP DMR	M-DMP	MIU	[48]

7.5.7.9 Guidelines for the encapsulation of H.263 streams

7.5.7.9.1 MT RTP H.263 RTP payload

If H.263 streams are transmitted over RTP, then the RTP payload format H263-2000 as defined in [46] and [36] must be used with the constraints listed in 7.5.7.9.2 through 7.5.7.9.4 below.

M	A	DMS +PU+	M-DMS	n/a	[46] [36]

7.5.7.9.2 MT RTP H.263 profile and level

A serving endpoint must specify the profile and level in the a=fmtp line of SDP.

M	A	DMS +PU+	M-DMS	n/a	[36]

The H.263 profile and level included in the a=fmtp line must follow the H263-2000 MIME media type specified in RFC 3555.

7.5.7.9.3 MT RTP H.263 frame size attribute at SDP media level

7.5.7.9.3.1 A serving endpoint must specify the a=framesize attribute at the SDP media level for each H.263 stream in SDP.

The syntax is defined as:

- a=framesize = "a=framesize:" payload-type SP width "-" height;
- payload-type = 1*DIGIT; (must be between 0 and 127);
- width = 1*DIGIT;
- height = 1*DIGIT.

Example:

- a=framesize:96 176-144

M	A	DMS +PU+	M-DMS	n/a	n/a

7.5.7.9.3.2 Receiving endpoints may support the a=framesize SDP attribute.

O	A	DMP DMR	M-DMP	MIU	n/a

7.5.7.9.4 MT RTP H.263 ADU usage

An ADU must be a complete video slice. Each RTP packet must carry a single ADU.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[46]

7.5.7.10 Guidelines for the encapsulation AMR streams

7.5.7.10.1 MT RTP AMR RTP payload

If AMR streams are transmitted over RTP, then the RTP payload format RFC 3267 must be used as specified in 7.5.7.10.2 to 7.5.7.10.4, inclusive.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[42]

7.5.7.10.2 MT RTP AMR RTP encapsulation

7.5.7.10.2.1 A serving endpoint must support encapsulation of one or more AMR speech frames into a single RTP packet and must include maxptime attribute in SDP.

M	A	DMS +PU+	M-DMS	n/a	[42]

Section 8.1 of [42] explains the SDP maxptime attribute.

7.5.7.10.2.2 The recommended value for the SDP maxptime attribute is 200 ms.

S	A	DMS +PU+	M-DMS	n/a	[42]

7.5.7.10.3 MT RTP AMR RTP interleaving

7.5.7.10.3.1 A Serving Endpoint may support interleaving.

O	A	DMS +PU+	M-DMS	n/a	[42]

7.5.7.10.3.2 If a serving endpoint supports interleaving then it must apply interleaving when the `rtp-amr-deint-buf-cap.dlna.org` header is present in the RTSP DESCRIBE request.

M	A	DMS +PU+	M-DMS	n/a	[42]

7.5.7.10.3.3 A serving endpoint must set the value of the interleaving parameter in the `a=fmtp` line of the SDP equal to or less than the value of the `rtp-amr-deint-buf-cap.dlna.org` header in the RTSP DESCRIBE request.

M	A	DMS +PU+	M-DMS	n/a	[42]

Subclause 8.1 of [42] explains the SDP interleaving parameter.

The value of the SDP interleaving parameter in `a=fmtp` can be 0.

7.5.7.10.3.4 A receiving endpoint may support interleaving.

O	A	DMP DMR	M-DMP	MIU	[42]

7.5.7.10.3.5 If a receiving endpoint supports interleaving then it must include the `rtp-amr-deint-buf-cap.dlna.org` header in the DESCRIBE request.

The syntax of this header is specified as follows:

- `rtp-amr-deint-buf-cap-line` = "rtp-amr-deint-buf-cap.dlna.org" *LWS ":" *LWS 1*DIGIT

The value of the `rtp-amr-deint-buf-cap.dlna.org` header is interpreted identically to the "interleaving" SDP parameter, specified in [42].

M	A	DMP DMR	M-DMP	MIU	[42]

Subclause 8.1 of [42] explains the SDP "interleaving" parameter.

7.5.7.10.4 MT RTP AMR RTP decapsulation

7.5.7.10.4.1 A receiving endpoint must be able to decapsulate an RTP packet having one or more AMR speech frames.

M	A	DMP DMR	M-DMP	MIU	[42]

7.5.7.10.4.2 A receiving endpoint should be able to support values of the SDP maxptime attribute for at least 200 ms.

S	A	DMP DMR	M-DMP	MIU	[42]

Subclause 8.1 of [42] explains maxptime attribute.

7.5.7.11 Guidelines for the encapsulation AMR-WBplus streams

7.5.7.11.1 MT RTP payload format for AMR-WBplus streams

If AMR-WBplus streams are transmitted over RTP, then the RTP payload format [37] must be used as specified in 7.5.7.11.2 to 7.5.7.11.6 inclusive.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[37]

7.5.7.11.2 MT RTP AMR-WBplus encapsulation

7.5.7.11.2.1 A serving endpoint must support encapsulation of one or several AMR-WBplus frames into a single RTP packet and must include maxptime attribute in SDP.

M	A	DMS +PU+	M-DMS	n/a	[37]

7.5.7.11.2.2 The recommended value for the SDP maxptime attribute is 200 ms.

S	A	DMS +PU+	M-DMS	n/a	[37]

7.5.7.11.3 MT RTP AMR-WBplus basic mode

7.5.7.11.3.1 A serving endpoint must support basic mode.

M	C	DMS +PU+	M-DMS	n/a	[37]

7.5.7.11.3.2 A Receiving Endpoint must support basic mode.

M	R	DMP DMR	M-DMP	MIU	[37]

7.5.7.11.4 MT RTP AMR-WBplus interleaving mode

7.5.7.11.4.1 A serving endpoint may support interleaving mode.

O	L	DMS +PU+	M-DMS	n/a	[37]

This is limitation to [37] that mandates to support both basic and interleaving modes.

7.5.7.11.4.2 If a serving endpoint supports interleaving mode then it must use interleaving when the `rtp-amrwbplus-deint-buf-cap.dlna.org` header is present in the RTSP DESCRIBE request.

M	R	DMS +PU+	M-DMS	n/a	[37]

7.5.7.11.4.3 A serving endpoint must set the value of the "interleaving" parameter in the `a=fmtp` line of the SDP equal to or less than the value of the `rtp-amrwbplus-deint-buf-cap.dlna.org` header in the RTSP DESCRIBE request.

M	R	DMS +PU+	M-DMS	n/a	[37]

Subclauses 4.4 and 7.2.1 of [37] explains usage of SDP parameters for interleaving.

7.5.7.11.4.4 If a receiving endpoint supports interleaving mode then it must include `rtp-amrwbplus-deint-buf-cap.dlna.org` header in the DESCRIBE request.

The syntax of this header is specified as follows:

- `rtp-amrwbplus-deint-buf-cap-line` = "rtp-amrwbplus-deint-buf-cap.dlna.org" *LWS ":" *LWS 1*DIGIT

The value of this header is interpreted identically to the "interleaving" SDP parameter, specified in [37].

M	R	DMP DMR	M-DMP	MIU	[37]

The value of interleaving parameter must be greater than zero (see 7.2.1 of [37]). If it is not present, interleaving mode must not be used,

Subclauses 4.4 and 7.2.1 of [37] explains usage of SDP parameters for interleaving.

7.5.7.11.5 MT RTP AMR-WBplus RTP decapsulation

7.5.7.11.5.1 A receiving endpoint must be able to decapsulate an RTP packet having one or more AMR-WBplus frames.

M	A	DMP DMR	M-DMP	MIU	[37]

7.5.7.11.5.2 A receiving endpoint should be able to support values of the SDP maxptime attribute for at least 200 ms.

S	A	DMP DMR	M-DMP	MIU	[37]

7.5.7.11.6 MT RTP AMR-WBplus channels

A receiving endpoint that is only capable of playout of monophonic audio must still accept signals originally encoded and transmitted as stereo.

M	C	DMS +PU+	M-DMS	n/a	[37]

The AMR-WBplus decoder has the capability of stereo to mono downmixing as part of the decoding process.

7.5.8 RTP media transport – RTSP for control of RTP streams

7.5.8.1 RTP media transport – RTSP for control of RTP streams – General

The real time streaming protocol (RTSP) provides a standard protocol for controlling media streams, including starting, stopping and pausing. Media operations such as fast forward scan, slow backward scan, etc. can also be implemented in a standard way using RTSP. RTSP is the required protocol for controlling streams for RTP transport.

The session description protocol (SDP) provides a standard method of communicating RTP session parameters. It is used as the format of the description of the media that is sent from the serving endpoint to the receiving endpoint in the response to the RTSP DESCRIBE method.

The DLNA RTSP and SDP signaling in RTSP and SDP signalling for DLNA (7.5.8.2.1 to 7.5.8.2.80) specifies a minimal strict subset of RTSP (RFC 2326 [33]) and SDP (RFC 2327 [34]) to be implemented by DLNA serving endpoints and receiving endpoints.

7.5.8.2 RTP media transport – RTSP/SDP Protocols, RTSP and SDP – Signalling for DLNA

7.5.8.2.1 MT RTP RTSP support

A DLNA device must implement RTSP version 1.0 as the mandatory media transport control protocol as described in Appendix D of RFC 2326 with constraints and extensions defined in subsequent entries of this table. Specifically serving endpoints must implement RTSP servers and receiving endpoints must implement RTSP clients.

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33]

RTSP controls the RTP media transport.

7.5.8.2.2 MT RTP RTSP SDP support

A DLNA device must implement SDP as the mandatory format used in the RTSP DESCRIBE method.

M	R	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[34]

SDP declares the properties of the media streams that may be present in the RTSP session.

7.5.8.2.3 MT RTP RTSP TCP transport

7.5.8.2.3.1 Both serving endpoint and receiving endpoint devices must use TCP to transport RTSP

M	L	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33]

RTSP/UDP usage is not allowed.

7.5.8.2.3.2 A serving endpoint should support persistent TCP connections. This means that the TCP connection should not be closed by serving endpoint until after all RTSP sessions using the TCP connection have been terminated.

S	C	DMS +PU+	M-DMS	n/a	[33]

7.5.8.2.3.3 A receiving endpoint should use a persistent TCP connection for each RTSP session. This means that the TCP connection should not be closed by receiving endpoint until after the RTSP session has been terminated.

S	C	DMP DMR	M-DMP	MIU	[33]

7.5.8.2.4 MT RTP RTSP pipelined requests

Serving endpoint must accept pipelined RTSP requests, and must respond to RTSP requests in the order that they were received.

M	C	DMS +PU+	M-DMS	n/a	[33]

7.5.8.2.5 MT RTP multiple RTSP sessions

Serving endpoints should support more than one simultaneous RTSP session.

S	C	DMS +PU+	M-DMS	n/a	[33]

It is a good practice for a serving endpoint to support multiple receiving endpoints simultaneously.

7.5.8.2.6 MT RTP RTSP session multiplexing

Receiving endpoint must use a separate TCP connection for each RTSP session.

M	L	DMP DMR	M-DMP	MIU	[33]

7.5.8.2.7 MT RTP receive RTSP requests from a serving endpoint

A receiving endpoint must support receiving RTSP requests from the serving endpoint over the same TCP connection that the receiving endpoint is using for sending requests to the serving endpoint.

M	C	DMP DMR	M-DMP	MIU	[33]

If a receiving endpoint does not implement support for an RTSP request type, it must respond to a serving endpoint with a 501 (not implemented) status code.

M	C	DMP DMR	M-DMP	MIU	[33]

7.5.8.2.8 MT RTP indication of supported RTSP version

7.5.8.2.8.1 Serving endpoints and receiving endpoints should specify the highest RTSP version number that they are compliant with when sending a RTSP response regardless of which RTSP version number was specified in the corresponding RTSP request.

S	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33]

7.5.8.2.8.2 If a serving endpoint or a receiving endpoint receives a RTSP request that specifies a higher version of RTSP than is supported, then it must respond with status code 505 (RTSP version not supported).

M	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33]

7.5.8.2.8.3 Serving endpoints and receiving endpoints must accept RTSP responses that specify a RTSP version number higher than the highest supported RTSP version.

M	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33]

7.5.8.2.9 MT RTP tolerate unknown RTSP headers and SDP attributes

Receiving endpoints and serving endpoints must be tolerant of unknown RTSP headers and SDP attributes.

Tolerant behaviour is defined as being able to successfully "parse and interpret" or "parse and ignore" the RTSP headers, their values and SDP attributes and their values.

M	R	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33] [34]

This guideline addresses forward compatibility and allows for broader interoperability with implementations that employ transport layer vendor extensions by way of RTSP headers.

7.5.8.2.10 MT RTP RTSP case-sensitivity

Names of RTSP methods must be treated as case sensitive tokens, while the RTSP headers must be treated as case-insensitive tokens.

M	C	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[33]

This is normative according to the RTSP specification.

7.5.8.2.11 MT RTP RTSP required requests A serving endpoint must support receiving the following RTSP requests:

- DESCRIBE;
- SETUP;
- PLAY;
- OPTIONS;
- TEARDOWN.

A serving endpoint need not support receiving any other RTSP requests.

M	A	DMS +PU+	M-DMS	n/a	[33]

7.5.8.2.11.2 A receiving endpoint must not depend on the support of any RTSP request not listed in 7.5.8.2.11.1.

M	A	DMP DMR	M-DMP	MIU	[33]

7.5.8.2.11.3 A receiving endpoint must support sending the following RTSP requests:

- DESCRIBE;
- SETUP;
- PLAY;
- OPTIONS;
- TEARDOWN.

M	A	DMP DMR	M-DMP	MIU	[33]

7.5.8.2.12 MT RTP DLNA media operations summary

The DLNA streaming media operations (in Table 22) must be implemented as defined in the following guidelines:

- Play: 7.5.8.2.35;
- Stop: 7.5.8.2.54;

- Pause: 7.5.8.2.46;
- Pause-release: 7.5.8.2.48;
- Seek: 7.5.8.2.38;
- Fast Forward Scan: 7.5.8.2.42;
- Slow Forward Scan: 7.5.8.2.42;
- Fast Backward Scan: 7.5.8.2.42;
- Slow Backward Scan: 7.5.8.2.42;
- Streaming Download: n/a.

M	A	DMP DMR	M-DMP	MIU	n/a

7.5.8.2.13 MT RTP CSeq header

A serving endpoint must include the CSeq header in all RTSP responses and in any RTSP request that it sends.

The CSeq header in requests sent by the serving endpoint must be independent of the CSeq header in requests sent by the receiving endpoint.

M	R	DMS +PU+	M-DMS	n/a	[33]

7.5.8.2.14 MT RTP supported header

The supported header field must comply with the following syntax:

- supported-line = "Supported" *LWS ":" *LWS [feature-tag *(*LWS "," *LWS feature-tag)];
- feature-tag = token.

Examples:

- supported: dlina.announce;
- supported: dlina.announce, rtsp.basic.

The RTSP header name is the supported string and the header value is zero or more comma-separated tokens. Both header name and value are treated as case insensitive strings.

It should be noted that the supported header may be used by serving endpoints and receiving endpoints issuing an RTSP request on non-DLNA content.

M	A	DMS DMP DMR +PU+	M-DMS M-DMP	MIU	[41]

The supported header is currently defined for the SIP protocol (see 20.37 of [41]).

7.5.8.2.15 MT RTP Event-Type.dlna.org header

The Event-Type.dlna.org header must comply with the following syntax:

- event-type-line = "Event-Type.dlna.org" *LWS ":" *LWS event code;
- event-code = 4DIGIT; 0-9999.