

CONTENTS

7			
8	Foreword		6
9	Introduction		7
10	1 Scope.....		8
11	2 Normative References		8
12	3 Definitions and Abbreviations.....		8
13	3.1 Definitions		8
14	3.2 Abbreviations		10
15	4 Conformance.....		10
16	5 Device Type Overview		10
17	6 Entity Device Type.....		11
18	7 Functional Device Type		12
19	7.1 Summary on Functional Device Type		12
20	7.2 Basic Functional Device Type		12
21	7.2.1 Media Server		12
22	7.2.1.1 Overview		12
23	7.2.1.2 Device Type		12
24	7.2.1.3 The Interaction Framework and Flow.....		12
25	7.2.1.4 Sub-service and the Mandatory Interface Definitions.....		17
26	7.2.2 Media Player.....		18
27	7.2.2.1 Overview		18
28	7.2.2.2 Device Type		19
29	7.2.2.3 The Interaction Framework and Flow.....		19
30	7.2.2.4 Sub-service and the Mandatory Interface Definitions.....		23
31	7.2.3 Media Recorder		24
32	7.2.3.1 Overview		24
33	7.2.3.2 Device Type		25
34	7.2.3.3 The Interaction Framework and Flow.....		25
35	7.2.3.4 Sub-service and the Mandatory Interface Definitions.....		26
36	7.2.4 File Server		27
37	7.2.4.1 Overview		27
38	7.2.4.2 Device Type		27
39	7.2.4.3 The Interaction Framework and Flow.....		27
40	7.2.4.4 Sub-service and the Mandatory Interface Definitions.....		30
41	Bibliography		32
42			

43		
		FIGURES
44	Figure 1 - Interaction Flow between Media Server and Controller in PULL Mode	13
45	Figure 2 - Interaction Flow between Media Server and Controller in PUSH Mode.....	15
46	Figure 3 - Interaction Flow between Media Player and Controller in PULL Mode.....	20
47	Figure 4 - Interaction Flow between Media Player and CTR in PUSH Mode	22
48	Figure 5 - Interaction Flow between Media Recorder and CTR	25
49	Figure 6 - Interaction Flow between FileServer and FileClient	28

50

51		
		TABLES
52	Table 1 - Basic Entity Device Type List.....	11
53	Table 2 – Basic Functional Device Type List.....	12
54	Table 3 – Sub-service List of Media Server.....	17
55	Table 4 - Mandatory Interface List of Media Server Service.....	18
56	Table 5 – Sub-service List of Media Player	23
57	Table 6 - Mandatory Interface List of Media Player	24
58	Table 7 – Sub-service List of Media Recorder.....	26
59	Table 8 - Mandatory Interface List of Media Recorder	27
60	Table 9 – Sub-service List of File Server	30
61	Table 10 - Mandatory Interface List of File Server.....	31

62

Foreword

63 ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission) form the
64 specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the
65 development of International Standards through technical committees established by the respective organization to
66 deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual
67 interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take
68 part in the work.

- 69 1) In the field of information technology, ISO and IEC have established a joint technical committee ISO/IEC JTC 1.
- 70 2) The formal decisions or agreements of ISO and IEC on technical matters express, as nearly as possible, an
71 international consensus of opinion on the relevant subjects since each technical committee has representation from
72 all interested National Committees.
- 73 3) The documents produced have the form of recommendations for international use and are published in the form of
74 standards, technical reports, technical specifications or guides and they are accepted by the National Committees
75 in that sense.
- 76 4) In order to promote international unification, ISO National Members and IEC National Committees undertake to
77 apply ISO and IEC International Standards transparently to the maximum extent possible in their national and
78 regional standards. Any divergence between the IEC standard and the corresponding national or regional standard
79 shall be clearly indicated in the latter.
- 80 5) ISO and IEC provide no marking procedure to indicate its approval and cannot be rendered responsible for any
81 equipment declared to be in conformity with one of its standards.
- 82 6) Attention is drawn to the possibility that some of the elements of this standard may be the subject of patent rights.
83 The IEC shall not be held responsible for identifying any or all such patent rights.

84 ISO/IEC 14543-5-5 was prepared by Joint Technical Committee 1 of ISO/IEC, Information technology, Subcommittee
85 SC 25, Interconnection of Information Technology Equipment.

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

87

Introduction

88 ISO/IEC 14543-5, Intelligent Grouping and Resource Sharing for HES (IGRS), is divided into six
89 parts:

90 ➤ **ISO/IEC 14543 - Part 5-1: Core Protocol**

91 • Specifies the TCP/IP protocol stack as the basis and the HTTP protocol as the message-
92 exchanging framework among devices.

93 • Defines a series of device and service interaction/invocation standards, including device
94 and service discovery protocol, device and service description, service invocation,
95 security mechanisms, etc.

96 • Specifies core protocols for a type of home network that supports streaming media and
97 other high-speed data transport within a home.

98 ➤ **ISO/IEC 14543 - Part 5-2: Application Profile**

99 • Based on the IGRS Core Protocol.

100 • Defines a device and service interaction mechanism, as well as application interfaces
101 used in IGRS Basic Applications.

102 ➤ **ISO/IEC 14543 - Part 5-3: Basic Application**

103 • Includes an IGRS basic application list.

104 • Defines a basic application framework.

105 • Addresses operation specifics (device grouping, service description template, etc.),
106 function definitions, and service invocation interfaces.

107 ➤ **ISO/IEC 14543 - Part 5-4: Device Validation**

108 • Defines a standard method to validate an IGRS-compliant device.

109 ➤ **ISO/IEC 14543 - Part 5-5: Device Types**

110 • Defines IGRS Device types used in IGRS applications.

111 ➤ **ISO/IEC 14543 - Part 5-6: Service Types**

112 • Defines basic service types used in IGRS applications.

113 **Information technology –**
114 **Home Electronic System (HES) Architecture – Intelligent Grouping and**
115 **Resource Sharing for HES Class 2 & Class 3 – Part 5: Device Type**

116 **1 Scope**

117 This part of the ISO/IEC 14543-5 specifies the device type which conforms to the Part 5-1: Core
118 Protocol and Part 5-2: Application Profile.

119 This part of the ISO/IEC 14543-5 is applicable to all devices that are operating in IGRS network.
120 In addition, all IGRS devices shall conform to this standard.

121 **2 Normative References**

122 The following standards and specifications contain provisions that, through reference in this text,
123 constitute normative provisions of this standard. For all referenced articles with dates, the
124 revised edition after that date shall not be applicable to this document. However, the reader is
125 encouraged to consider whether to use the latest edition of such articles. For all referenced
126 articles without dates, the latest edition of such articles shall be applicable to this document.

127 ISO/IEC 14543-5-1, *Information Technology - Information Device Intelligent Grouping and*
128 *Resource Sharing Specification Part 1: Core Protocol*

129 **3 Definitions and Abbreviations**

130 **3.1 Definitions**

131 The following terms are used in this standard and commonly used in other industry publications.
132 As used in this standard, they have the following meanings:

133 **3.1.1**

134 **Centralised Device Group**

135 Set of IGRS Devices with one IGRS Device acting as the master. The master is responsible for
136 managing the setup, for dismissing a Device Group, and for processing a join request from other
137 devices. The master device and other IGRS Devices in such a Device Group form a centralised
138 or master-slave relationship.

139 **3.1.2**

140 **Client Identifier**

141 Unique identifier associated with a Client on an IGRS Device to which that Client belongs

142 **3.1.3**

143 **Device Group**

144 Multiple IGRS Devices that are organised into a logical group through the device group
145 management mechanism in the IGRS specification. Each IGRS Device in a Device Group follows
146 common interaction rules. Two types of Device Groups are defined: peer-to-peer Device Group
147 and centralised (master-slave) Device Group.

148 **3.1.4**

149 **Device Identifier**

150 Globally unique device identifier associated with one IGRS Device

- 151 **3.1.5**
152 **Device Pipe**
153 Channel used to transfer device interaction messages. This channel is set up through the pipe
154 setup mechanism in the IGRS Specification
- 155 **3.1.6**
156 **Device Type**
157 Identifier that indicates the physical and functional characteristic shown by an IGRS device
- 158 **3.1.7**
159 **Entity Device Type**
160 Identifier that indicates the physical form of a device
- 161 **3.1.8**
162 **Functional Device Type**
163 Identifier that indicates the functional characteristic of a device
- 164 **3.1.9**
165 **IGRS Client**
166 Application that draws upon the services of one or more connected IGRS Devices. Multiple client
167 instances can exist on a network at the same time.
- 168 **3.1.10**
169 **IGRS Device**
170 Information device that conforms to the IGRS specification
- 171 **3.1.11**
172 **IGRS Service**
173 Sharable resource encapsulated in an IGRS Device by implementing application interfaces and
174 providing services for other IGRS Devices. An IGRS Service has an invocation interface that
175 meets the requirements of the IGRS specification. These invocation interfaces are described and
176 announced on the network through the IGRS Service Description Specification.
- 177 **3.1.12**
178 **IGRS User**
179 owner of an IGRS Device and Client
- 180 **3.1.13**
181 **Mandatory Interface**
182 Service interface that shall be implemented by an IGRS device of some functional device type
- 183 **Peer-to-Peer Device Group**
184 Set of IGRS Devices with where each IGRS Device in this set has a peer-to-peer relationship
185 with each other
- 186 **3.1.14**
187 **Service Identifier**
188 Unique identifier assigned to a service provided by a specific IGRS Device. Note that the same
189 type of service may be provided by multiple IGRS Devices within the same network. Each
190 instance of a service has a unique service identifier on the IGRS Device providing that service
- 191 **3.1.15**
192 **Service Type**
193 Category of IGRS Service defined according to the set of resources encapsulated. The Service
194 Type enables service applications in the same category to have common invocation interfaces

195 **3.1.16**
196 **Sub-service**
197 Specific set of service which is a part of some functional device types

198 **3.1.17**
199 **User Identifier**
200 Identifier of an IGRS user

201 **3.2 Abbreviations**

CIS	Content Index Service
CMS	Connection Management Service
CTR	Controller
FAMS	FileAccessManagement Service
FC	FileClient
FCMS	FileConnectionManagement Service
FS	FileServer
MP	Media Player
MPTMS	Media Player Transport Management Service
MR	Media Recorder
MS	Media Server
MSTMS	Media Server Transport Management Service
RMS	Rendering Management Service

202 **4 Conformance**

203 For conformance to this international standard the following applies:

- 204 • IGRS device type classification model shall conform to the Device Type Overview
205 specification described in Clause 5.
- 206 • Entity device type shall conform to specifications described in Clause 6.
- 207 • Functional device type shall conform to the specification defined in Clause 7.

208 **5 Device Type Overview**

209 IGRS device type can be divided into two types by concept. One type is the Entity Device Type;
210 the other is Functional Device Type. The Entity Device Type is used to indicate the physical form
211 of a device, such as PC, TV and so on. The Functional Device Type is used to indicate the
212 common functional characteristic of devices, for example, Media Server, Media Player, etc.

213 The identifiers of the Entity Device Type and Functional Device Type of IGRS devices shall
214 conform to the definition of device type identifier in ISO/IEC 14543-5-1.

215 The syntactic definition of the device type identifier is defined as follows:

216 $\langle \text{IGRSSDeviceTypeURN} \rangle ::= \text{urn} : \langle \text{IGRSNS} \rangle : \langle \text{IGRSSingleType} \rangle | \langle \text{IGRSTypeList} \rangle$
217 $\langle \text{IGRSNS} \rangle ::= \langle \text{IGRSNS} \rangle ::= \text{IGRS} : \text{Device} : \text{DeviceType}$

218 When describing a single device type, the syntactic definition is:

219 $\langle \text{IGRSSDeviceTypeURN} \rangle ::= \text{urn} : \langle \text{IGRSNS} \rangle : \langle \text{IGRSSingleType} \rangle$
220 $\langle \text{IGRSSingleType} \rangle ::= \langle \text{NAME} \rangle$

221 When describing a multi-device type, the syntactic definition is:

222 $\langle \text{IGRSSDeviceTypeURN} \rangle ::= \text{urn} : \langle \text{IGRSNS} \rangle : \langle \text{IGRSTypeList} \rangle$
223 $\langle \text{IGRSTypeList} \rangle ::= \langle \text{IGRSSingleType} \rangle * \langle \text{IGRSTypeVal} \rangle$

224 <IGRSTypeVal>::=<ConnectionSign><IGRSSingleType>
 225 <NAME>::=1*16<URN chars>
 226 <URN chars>::=<trans>
 227 <trans>::=<upper>|<lower>|<number>|<other>
 228 <upper>::="A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L" | "M" | "N" | "O" |
 229 "P" | "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z"
 230 <lower>::="a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" | "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" | "s" |
 231 "t" | "u" | "v" | "w" | "x" | "y" | "z"
 232 <number>::="0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
 233 <other>::=- | . | _
 234 <ConnectionSign>::=;
 235 <URN chars> is case insensitive.

236 Every IGRS device should have one Entity Device Type and several Functional Device Types.

237 All device types related to the IGRS device shall be listed in the device type field of the device
 238 online advertisement message. However, only relevant types are listed in the device type field of
 239 device search and subscription messages.

240 When the both types are listed at the same time, the Entity Device Type shall be listed in front of
 241 the Functional Device Type.

242 **6 Entity Device Type**

243 The Entity Device Type is used to indicate the physical form of IGRS devices, and every device
 244 should have a unique Entity Device Type. The common Entity Device Types and the
 245 corresponding device type identifiers are defined in this section (see Table 1).

246 **Table 1 - Basic Entity Device Type List**

Entity Device Type Identifier	Field Explanation
urn:IGRS:Device:DeviceType:PC	IGRS PC
urn:IGRS:Device:DeviceType:NoteBook	IGRS Notebook
urn:IGRS:Device:DeviceType:PDA	IGRS PDA
urn:IGRS:Device:DeviceType:DC	IGRS Digital Camera
urn:IGRS:Device:DeviceType:DV	DV, conforming to IGRS's standards
urn:IGRS:Device:DeviceType:Mp3	IGRS MP3 Player
urn:IGRS:Device:DeviceType:Mp4	IGRS MP4 Player
urn:IGRS:Device:DeviceType:Mobilephone	IGRS Mobile Phone
urn:IGRS:Device:DeviceType:SetTopBox	IGRS Set-Top-Box
urn:IGRS:Device:DeviceType:TV	IGRS TV
urn:IGRS:Device:DeviceType:VCR	IGRS VCR
urn:IGRS:Device:DeviceType:DVDPlayer	IGRS CD/DVD Player
urn:IGRS:Device:DeviceType:DMA	IGRS Media Adaptor
urn:IGRS:Device:DeviceType:NAS	IGRS Network Attached Storage (NAS)
urn:IGRS:Device:DeviceType :UnknownType	IGRS unknown device type

247

248 **7 Functional Device Type**

249 **7.1 Summary on Functional Device Type**

250 The Functional Device Type identifies the functional form of IGRS devices. Every device can
251 have multiple Functional Device Types at the same time.

252 The Functional Device Type is defined by the sub-services and the mandatory interface set of
253 each sub-service.

254 This standard defines in details a series of basic Functional Device Types in the next few
255 sections. The descriptions include the related device type identifier, the sub-service set, the
256 mandatory interface set of each sub-service and the commonly used service invocation
257 procedure of the device type.

258 Table 2 defines a series of basic Functional Device Types.

259 **Table 2 – Basic Functional Device Type List**

Device Type Name	Functional Device Type Identifier	Field Explanation
MediaServer	urn:IGRS:Device:DeviceType:MediaServer	Device that provides media content. See section 7.2.1 for details.
MediaPlayer	urn:IGRS:Device:DeviceType:MediaPlayer	Device that plays media content. See section 7.2.2 for details.
MediaRecorder	urn:IGRS:Device:DeviceType:MediaRecorder	Device that records media content and uploads it to MediaServer. See section 7.2.3 for details.
FileServer	urn:IGRS:Device:DeviceType:FileServer	Device that provide file content. See section 7.2.4 for details.

260 **7.2 Basic Functional Device Type**

261 This section defines all of the basic functional devices listed above in details.

262 **7.2.1 Media Server**

263 **7.2.1.1 Overview**

264 MediaServer provides media content, and it includes CIS, CMS and optional MSTMS Services.

265 The functions of MS include:

- 266 a) Media content of MS provided to network can be browsed and searched by CTR, including
267 content format and supported transport protocol and so on;
- 268 b) Prepare for connection between MS and MP;
- 269 c) Media transport control (such as play, stop etc.);
- 270 d) Copy or move the media resource on other devices.

271 **7.2.1.2 Device Type**

272 The device type definition of the MediaServer is as follows:

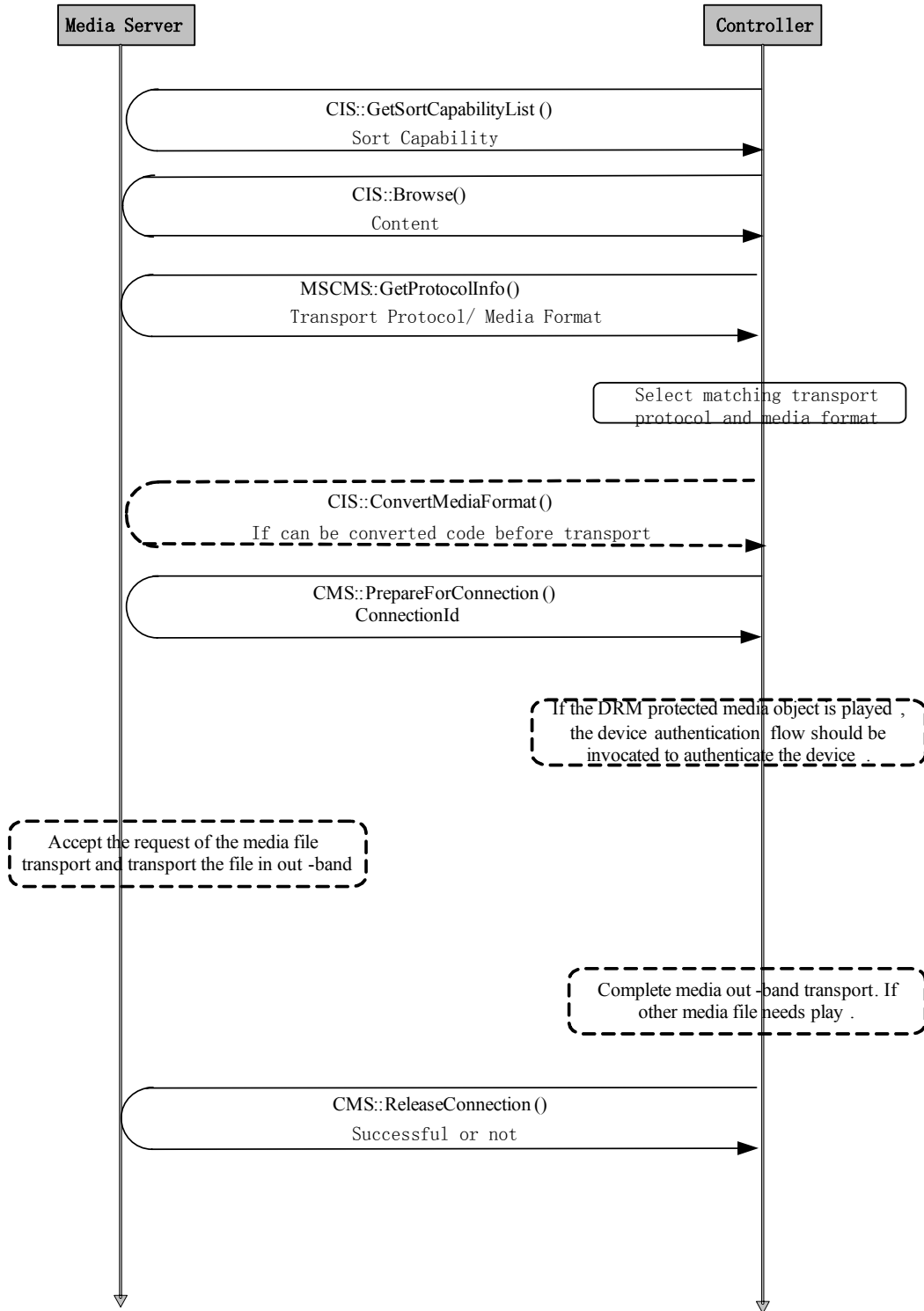
273 urn:IGRS:Device:DeviceType:MediaServer

274 **7.2.1.3 The Interaction Framework and Flow**

275 AV interaction model can be divided into two types: “Pull” and “Push” mode, depending on where
276 the TransportManagement Service of media is located (MS or MP). In “Pull” mode, media

277 TransportManagement Service is deployed in MP/MR. Media transport is managed by the sink
278 endpoint of media stream. In “Push” mode, media TransportManagement Service is deployed in
279 MS. Media transport is managed by the source endpoint of media stream.

280 In PULL mode, the interaction framework between Media Server and controller is as follows
281 (Figure 1):



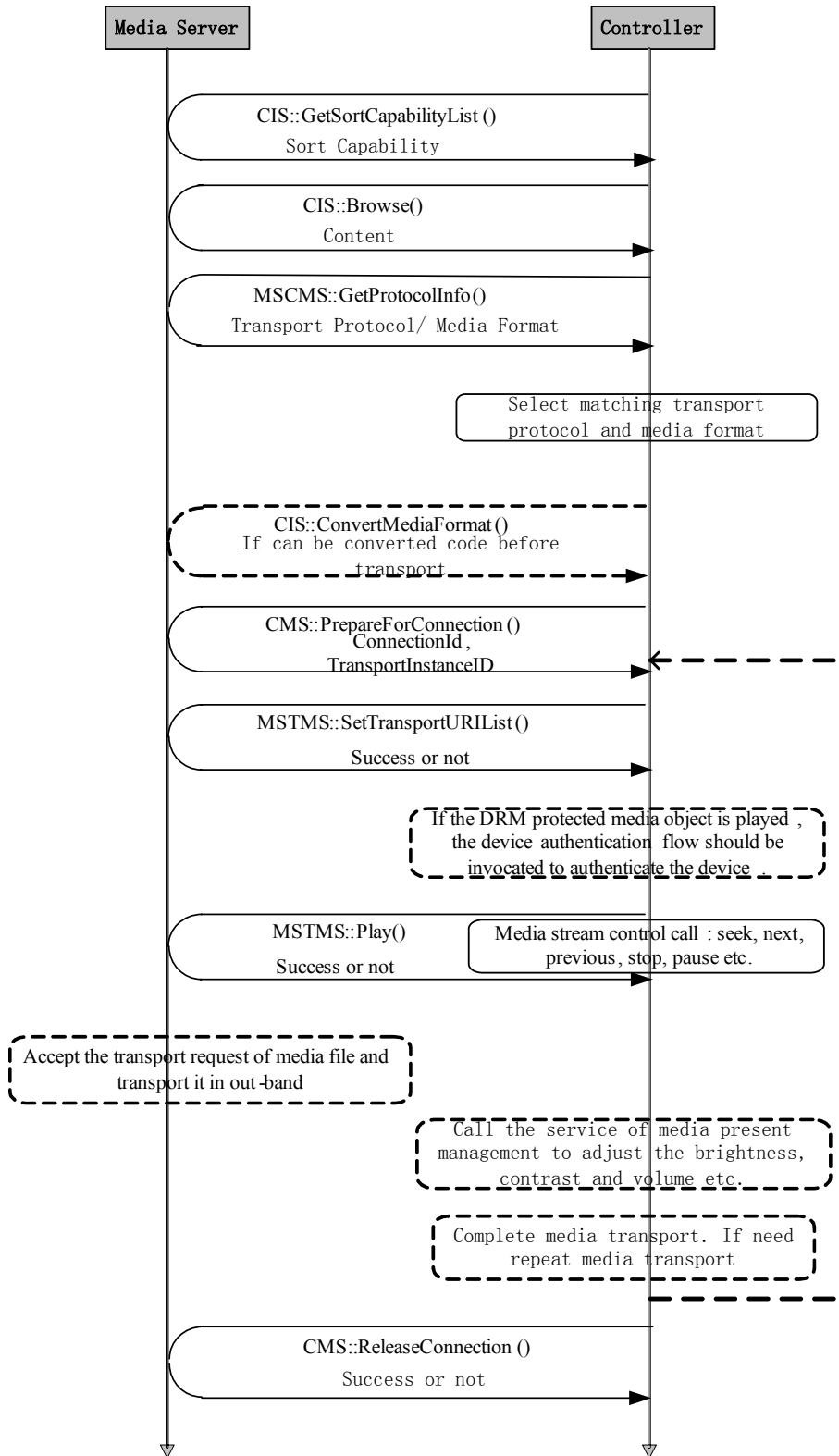
282

283 **Figure 1 - Interaction Flow between Media Server and Controller in PULL Mode**

284 The interaction flow between Media Server and Controller in PULL mode is as follows:

- 285 a) Through IGRS service discovery mechanism, CTR discovers the service on the IGRS device,
286 including CIS, MediaServer CMS;
- 287 b) Retrieve the sorting capability supported by the CIS of MS: Invoke the
288 CIS::GetSortCapabilityList() interface of CIS. The return value is the sorting capability
289 supported by CIS. If the return value is "*", it shall mean that CIS can sort all content
290 directory object attribute. If the return value is "DeviceName", it shall mean only device name
291 can be sorted.
- 292 c) Search, browse and locate the content need to be transported and played: Invoke
293 CIS::Browse() to browse media content. The return values include media name, media
294 format, corresponding connection management ID of the media content object, etc.
- 295 d) Retrieve the transport protocol, control protocol and media format supported by MS: Based
296 on the returned CMS ID of the corresponding media item in step c), CTR invokes the
297 CMS::GetProtocolInfo() interface of CMS to get the transport protocol, control protocol,
298 media format supported by CMS and the available IP address list of MS.
- 299 e) Matching process managed by CTR: Match the transport protocol, control protocol and media
300 format supported by MS and other devices (MP).
- 301 f) (Optional) Media transcoding: CTR invokes the CIS::ConvertMediaFormat() of CIS on MS to
302 ask if the media content format can be transcoded. CIS on MS determines whether it can
303 transcode the media format, and return the response to CTR.
- 304 g) Configure MS: Invoke CMS::PrepareForConnection() of CMS on MediaServer to notify MS
305 that a connection based on selected transport protocol and media format will be established.
306 The CMS of MediaServer returns to CTR the connection ID.
- 307 h) (Optional) Authenticate Controller: if DRM protected media is played, then IGRS DRM device
308 authentication process shall be invoked to authenticate the Controller.
- 309 i) Initiate media transport: Media Server receives the transport request of media file then
310 initiates the out-of-band transport of media. A HTTP 404 error is returned if the output media
311 format does not match the actual MS capability.
- 312 j) Release the connection: When the session between them is terminated, CTR shall invoke
313 the CMS::ReleaseConnection() of MediaServer CMS to disconnect the connection.

314 In PUSH mode, the interaction framework between Media Server and Controller is as follows
 315 (Figure 2):



316
317

318 **Figure 2 - Interaction Flow between Media Server and Controller in PUSH Mode**

319 Tthe interaction flow between Media Server and controller in PUSH mode is as follows:

- 320 a) Through IGRS service discovery mechanism, CTR discovers the service on the MediaServer
321 device, including CIS, MediaServer CMS and MediaServer TMS;
- 322 b) Retrieve the sorting capability supported by the CIS of MS: Invoke the
323 CIS::GetSortCapabilityList() interface of CIS. The return value is the sorting capability
324 supported by CIS. If the return value is "*", it shall mean that CIS can sort all content
325 directory object attribute. If the return value is "DeviceName", it shall mean only device name
326 can be sorted.
- 327 c) Search, browse and locate the content need to be transported and played: Invoke
328 CIS::Browse() to browse media content. The return values include media name, media
329 format, corresponding connection management ID of the media content object, etc.
- 330 d) Retrieve the transport protocol, control protocol and media format supported by MS: Based
331 on the returned CMS ID of the corresponding media item in step c), CTR invokes the
332 CMS::GetProtocolInfo() interface of CMS to get the transport protocol, control protocol,
333 media format supported by CMS and the available IP address list of MS.
- 334 e) Matching process managed by CTR: Match the transport protocol, control protocol and media
335 format supported by MS and other devices (MP).
- 336 f) (Optional) Media transcoding: CTR invokes the CIS::ConvertMediaFormat() of CIS on MS to
337 ask if the media content format can be transcoded. CIS on MS determines whether it can
338 transcode the media format, and return the response to CTR.
- 339 g) Configure MS: Invoke CMS::PrepareForConnection() of CMS on MediaServer to notify MS
340 that a connection based on selected transport protocol and media format will be established.
341 The CMS of MediaServer returns to CTR the connection ID. Meanwhile, the CMS of
342 MediaServer returns a transport instance ID to CTR based on selected transport protocol.
343 This ID is used to stream control (i.e. play, stop, seek etc.) by CMS of MediaServer.
- 344 h) Select transport content: CTR invokes the MSTMS::SetTransportURLList() to identify the
345 content to be transported. The input media list is the transport protocol matched by the
346 controller, control protocol, media info acquired by the controller, the connection identifier
347 and the available IP address list returned from Media Player etc.
- 348 i) (Optional) Authenticate Controller: if DRM protected media is played, then IGRS DRM device
349 authentication process shall be invoked to authenticate the Controller.
- 350 j) Initiate media transport: Use MSTMS to invoke transport management (e.g. play, stop, seek
351 etc.). A HTTP 404 error is returned if the output media format does not match the actual MS
352 capability.
- 353 k) (Optional) Change the current playing media object on the playing list: Invoke the Next or
354 Previous interface of MSTMS to play the next media object specified by the play mode or the
355 previous played media object. Then continue to play the media objects in turn according to
356 the play mode.
- 357 l) Release the connection: When the session between them is terminated, CTR shall invoke the
358 CMS::ReleaseConnection() of MediaServer CMS to disconnect the connection.

359 **7.2.1.4 Sub-service and the Mandatory Interface Definitions**

360 This section lists the sub-services of Media Server (Table 3), including required and optional
361 services, and their interface list (Table 4).

362 **Table 3 – Sub-service List of Media Server**

Service Name	Required /Optional	Service Type	Field Explanation
Content Index Service	Required	urn:IIGRS:Service:Service Type:ContentIndex:1	CIS allows CTR to discover and list the media content on MS so that CTR can retrieve the content information, including the name, creation date, size, format etc. of media content. This information can be used by CTR to determine whether these contents can be played on MP. The content directory structure supports nesting of sub-directory. See section 7.2 in Application Profile – AV Profile for details.
Connection Management Service	Required	urn:IIGRS:Service:Service Type:ConnectionManagement: 1	CMS is used to create and manage the connection between MS and MP. MS can support and manage several active connections at any one time by CMS. See section 7.3 in Application Profile – AV Profile for details.
Transport Management Service	Optional	urn:IIGRS:Service:Service Type:MediaServerTransportManagement:1	The optional MSTMS enables CTR to adjust and control the media stream transport on MS, such as play, pause, stop, seek etc. If this service exists, it means that the “Push” Mode is supported. Otherwise, it is not supported. See section 7.4 in Application Profile – AV Profile for details.

363

364

Table 4 - Mandatory Interface List of Media Server Service

Service Name	Mandatory Interface	Field Explanation
Content Index Service	GetSortCapabilityList	Retrieve the sorting list supported by CIS on MS
	GetContentUpdateId	Retrieve the value of content update ID of CIS on MS
	Browse	Return the browsed content directory object
Connection Management Service	GetProtocolInfo	Get the transport protocol supported by current device, including transport control protocol name, network port, content format, other protocol information and available IP address list on the MS.
	PrepareForConnection	Get the ConnectionId which is used to prepare and instantiate media data for sending and receiving, and TransportInstancelId which is used for media transport.
	ReleaseConnection	Notify device to disconnect the connection corresponding to ConnectionId and release resource.
	GetActiveConnectionIdList	Get current active connection ID list on MS or MP
Transport Management Service	SetTransportURIList	Set TransportURI variable. Specify the URI list of media content object to be transported by MSTMS instance.
	GetTransportInfo	Return the current transport state information of transport service instance
	Next	Set the next media object in the media list to be played. The selection mode is decided by the current media playing mode.
	Previous	Set the current media object as the previous played media object.
	Stop	Stop the current media object played by MSTMS. The current position should be reset in some devices.
	Play	Play media object in media list in sequence with specified playing mode (optional) and playing speed. (The starting point is decided by Offset). Playing is continued until the URI list on MS has been completely played or other interfaces have been invoked, such as stop etc. In addition, DRM transport should be established before playing copyrighted media content. The available content should include license and encrypted media content corresponding to the playing device. The encrypted media stream should be decrypted by DRM tool on MS or MP.
	Seek	Find the target position specified by input parameter in accordance to the input parameter Unit
	GetPlayURIList	Get MS URI list transported by specified transport service instance.
	GetAllMediaInfo	Get the detailed content information of media URI list transported by specified transport service instance.
	GetCurrentMediaInfo	Get media information transported by specified transport service instance.
GetCurrentPlayMode	Get the URI playing mode in the transport object list controlled by specified transport service instance	

365

366 7.2.2 Media Player**367 7.2.2.1 Overview**

368 IGRS MediaPlayer is used to transport and play media, and it includes CMS, RMS and optional
369 MPTMS Services. CMS allows CTR to get media format and transport control protocol supported
370 by MP. Similar with MS, MP can support multiple connections simultaneously, and enable CTR to
371 discover the active connection instances and its attributes. MPTMS enables CTR to adjust and

372 control the transport speed of media stream on MP. RMS gives CTR the ability to control the
373 media rendering, such as volume, contrast, brightness etc.

374 **7.2.2.2 Device Type**

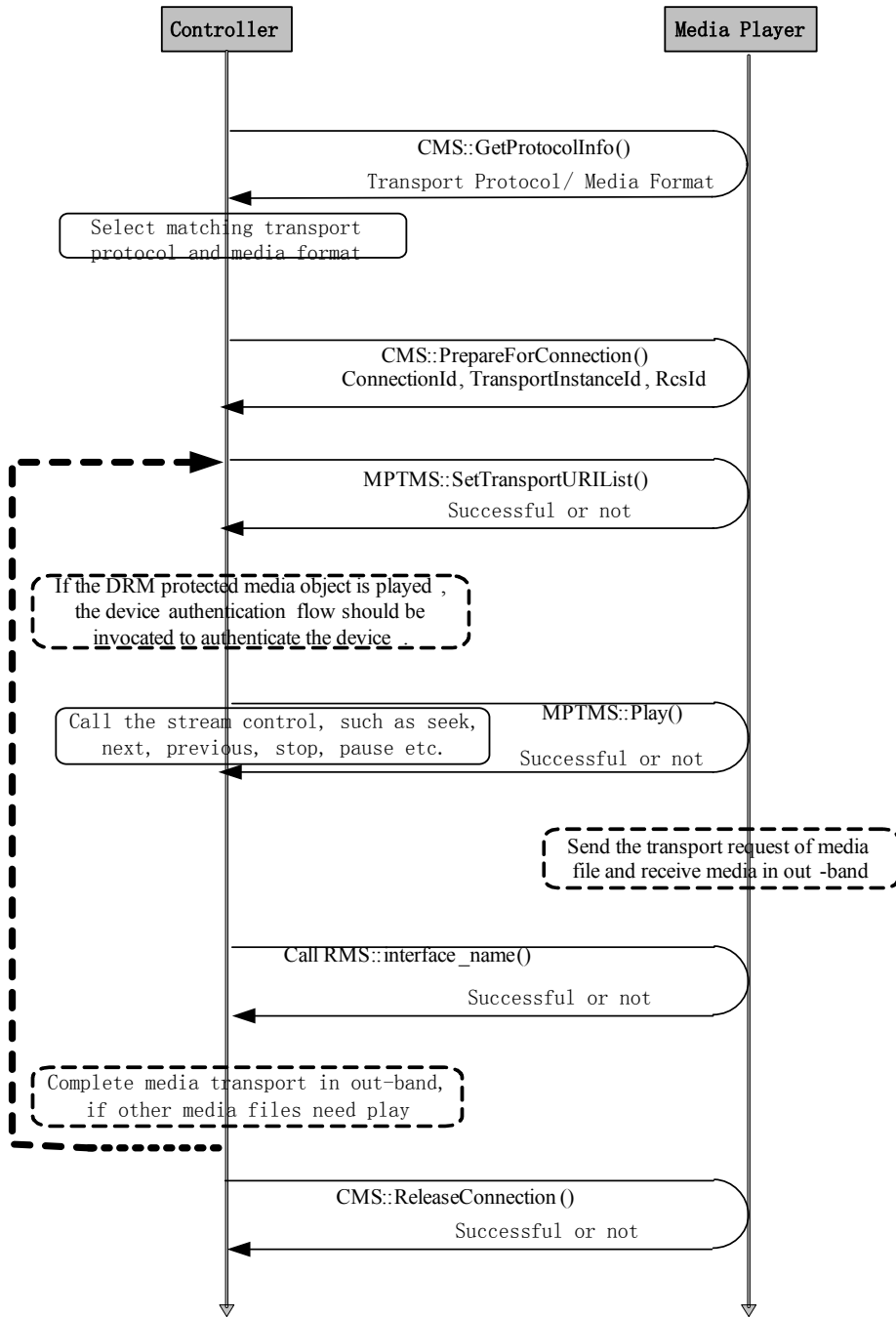
375 The device type definition of the MediaPlayer is as follows:

376 urn:IGRS:Device:DeviceType:MediaPlayer

377 **7.2.2.3 The Interaction Framework and Flow**

378 AV interaction model can be divided into two types: "Pull" and "Push" mode, depending on where
379 the TransportManagement Service of media is located (MS or MP). In "Pull" mode, media
380 TransportManagement Service is deployed in MP/MR. Media transport is managed by the sink
381 endpoint of media stream. In "Push" mode, media TransportManagement Service is deployed in
382 MS. Media transport is managed by the source endpoint of media stream.

383 In PULL mode, the interaction framework between Media Player and Controller is as follows
 384 (Figure 3):



385

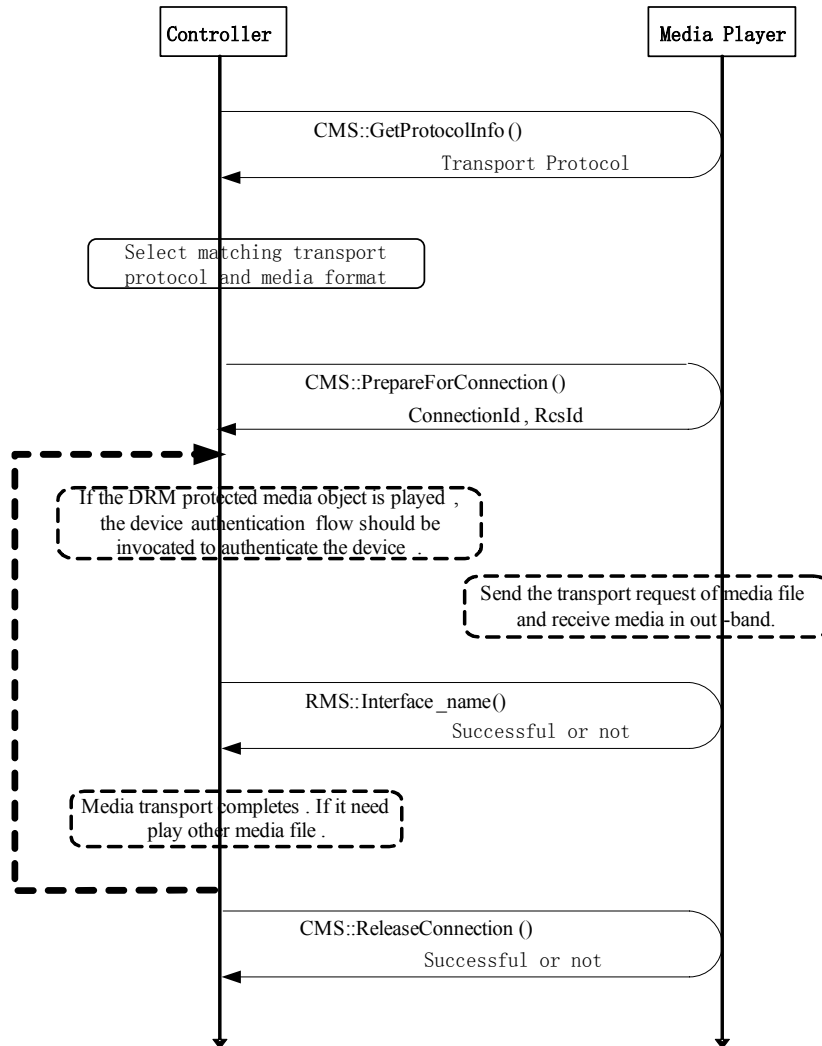
386 **Figure 3 - Interaction Flow between Media Player and Controller in PULL Mode**

387 The interaction flow between Media Player and Controller in PULL mode is as follows:

- 388 a) Through IGRS service discovery mechanism, CTR discovers the service on the MediaPlayer
- 389 device, including MediaPlayer CMS, TMS and RMS;
- 390 b) Retrieve the transport protocol, control protocol and media format supported by MP: CTR
- 391 invokes the CMS::GetProtocolInfo() interface of CMS to get the transport protocol, control
- 392 protocol, media format supported by CMS. If multiple CMSs exist on the device, then CTR

- 393 should invoke CMS::GetProtocolInfo interfaces of each CMS to get transport protocol, control
394 protocol and media format supported by each of these CMSs.
- 395 c) Matching process managed by CTR: Match the transport protocol, control protocol and media
396 format supported by MS and other devices (MP).
- 397 d) Configure MP: Invoke CMS::PrepareForConnection() of CMS on MediaPlayer to notify MP
398 that a connection based on selected transport protocol and media format will be established.
399 The CMS of Player returns to CTR the connection ID. Meanwhile, the CMS of MediaPlayer
400 returns a transport instance ID to CTR based on selected transport protocol. This ID is used
401 to stream control (i.e. play, stop, seek etc.) by CMS of MediaPlayer. In addition, CMS of
402 MediaPlayer returns a rendering management ID to CTR for control. When the
403 CMS::PrepareForConnection() of CMS on MediaPlayer is invoked, with the available IP
404 address list returned by CMS::GetProtocolInfo() of MS as input, the output is the IP address
405 list that can be connected.
- 406 e) Select transport content: CTR invokes the MPTMS::SetTransportURIList() to identify the
407 content to be transported, which is located in the available IP address list from step d).
- 408 f) (Optional) Authenticate Controller: if DRM protected media is played, then IGRS DRM device
409 authentication process shall be invoked to authenticate the Controller.
- 410 g) Initiate media transport: Use MPTMS to invoke transport management (e.g. play, stop, seek
411 etc.).
- 412 h) (Optional) Change the rendering characteristic of media object: Invoke RMS of MP to adjust
413 the rendering characteristic (e.g. brightness, contrast, volume, mute etc.)
- 414 i) (Optional) Change the current playing media object on the playing list: Invoke the Next or
415 Previous interface of MPTMS to play the next media object specified by the play mode or the
416 previous played media object. Then continue to play the media objects in turn according to
417 the play mode.
- 418 j) Release the connection: When the session between them is terminated, CTR shall invoke the
419 CMS::ReleaseConnection() of MediaPlayer CMS to disconnect the connection.

420 In PUSH mode, the interaction framework between Media Player and CTR is as follows (Figure
421 4):



422

423

Figure 4 - Interaction Flow between Media Player and CTR in PUSH Mode

424 The interaction flow between Media Player and CTR in PUSH mode is as follows:

- 425 a) Through IGRS service discovery mechanism, CTR discovers the service on the MediaPlayer
- 426 device, including MediaPlayer TMS and RMS;
- 427 b) Retrieve the transport protocol, control protocol and media format supported by MP: CTR
- 428 invokes the CMS::GetProtocolInfo() interface of CMS to get the transport protocol, control
- 429 protocol, media format supported by CMS. If multiple CMSs exist on the device, then CTR
- 430 should invoke CMS::GetProtocolInfo interfaces of each CMS to get transport protocol, control
- 431 protocol and media format supported by each of these CMSs.
- 432 c) Matching process managed by CTR: Match the transport protocol, control protocol and media
- 433 format supported by MS and other devices (MP, MR).
- 434 d) Configure MP: Invoke CMS::PrepareForConnection() of CMS on MediaPlayer to notify MP
- 435 that a connection based on selected transport protocol and media format will be established.
- 436 The CMS of Player returns to CTR the connection ID. In addition, CMS of MediaPlayer
- 437 returns a rendering management ID to CTR for control. When the
- 438 CMS::PrepareForConnection() of CMS on MediaPlayer is invoked, with the available IP
- 439 address list returned by CMS::GetProtocolInfo() of MS as input, the output is the IP address
- 440 list that can be connected.

- 441 e) (Optional) Authenticate Controller: if DRM protected media is played, then IGRS DRM device
- 442 authentication process shall be invoked to authenticate the Controller.
- 443 f) Initiate media transport: Use MPTMS to invoke transport management;
- 444 g) (Optional) Change the rendering characteristic of media object: Invoke RMS of MP to adjust
- 445 the rendering characteristic (e.g. brightness, contrast, volume, mute etc.)
- 446 h) Release the connection: When the session between them is terminated, CTR shall invoke the
- 447 CMS::ReleaseConnection() of MediaPlayer CMS to disconnect the connection.

448 **7.2.2.4 Sub-service and the Mandatory Interface Definitions**

449 This section lists the sub-services of Media Player (Table 5), including required and optional
450 services, and their interface list (Table 6).

451 **Table 5 – Sub-service List of Media Player**

Service Name	Required /Optional	Service Type	Field Explanation
Connection Management Service	Required	urn:IGRS:Service:ServiceType:ConnectionManagement:1	CMS is used to create and manage the connection between MS and MP. MS can support and manage several active connections at any one time by CMS. See section8.2 in Application Profile – AV Profile for details.
Transport Management Service	Optional	urn:IGRS:Service:ServiceType:MediaPlayerTransport Management:1	The optional MPTMS service enables CTR to adjust and control the transport of media stream on MP, such as play, pause, stop, seek etc. If this service exists, it means that the “Pull” Mode is supported. Otherwise, it is not supported. See section 8.3 in Application Profile – AV Profile for details.
Rendering Management Service	Required	urn:IGRS:Service:ServiceType:RenderingManagement :1	RenderingManagement Service allows CTR to control media playing, such as volume, contrast, brightness etc. Multiple active instances of rendering control can be supported, such as the “picture-in-picture” function in TV. New RMS instance is created by PrepareForConnection of CMS on MP. This service is only provided by MP. See section 8.4 in Application Profile – AV Profile for details.

452

453

Table 6 - Mandatory Interface List of Media Player

Service Name	Mandatory Interface	Field Description
Connection Management Service	GetProtocolInfo	Get the transport protocol supported by current device, including transport control protocol name, network port, content format, other protocol information and available IP address list on the MS.
	PrepareForConnection	Get the ConnectionId which is used to prepare and instantiate media data for sending and receiving, and TransportInstanceid which is used for media transport.
	ReleaseConnection	Notify device to disconnect the connection corresponding to ConnectionId and release resource.
	GetActiveConnectionIdList	Get current active connection ID list on MS or MP.
Transport Management Service	SetTransportURLList	Set TransportURI variable. Specify the URI list of media content object to be transport by MPTMS instance.
	GetTransportInfo	Get current transport state information of transport service instance.
	Play	Play media object in media list in sequence with specified playing mode (optional) and playing speed. (The starting point is decided by Offset). Playing is continued until the URI list on MS has been completely played or other interfaces have been invoked, such as stop etc. In addition, DRM transport should be established before playing copyrighted media content. The available content should include license and encrypted media content corresponding to the playing device. The encrypted media stream should be decrypted by DRM tool on MS or MP.
	Next	Set the next media object in the media list to be played. The selection mode is decided by the current media playing mode.
	Previous	Set the current media object as the previous played media object.
	Stop	Stop the current media object played by MPTMS. The current position should be reset in some devices.
	Seek	Find the target position specified by input parameter in accordance to the input parameter Unit
	GetPlayURLList	Get MS URI list transported by appointed transport service instance.
	GetAllMediaInfo	Get the detailed content information of media URI list transported by specified transport service instance.
	GetCurrentMediaInfo	Get media information transported by specified transport service instance.
GetCurrentPlayMode	Get the URI playing mode in the transport object list controlled by specified transport service instance	
Rendering Management Service	ListPresets	Get current rendering control state variable list
	SelectPresets	Recover rendering control state variable to preset value

454

455 **7.2.3 Media Recorder**456 **7.2.3.1 Overview**

457 MR transports the recorded of media stream to MS. Its configuration is similar to that of MP in
458 that both ConnectionManagement Service and MediaPlayerTransportManagement Service are
459 provided.

460 **7.2.3.2 Device Type**

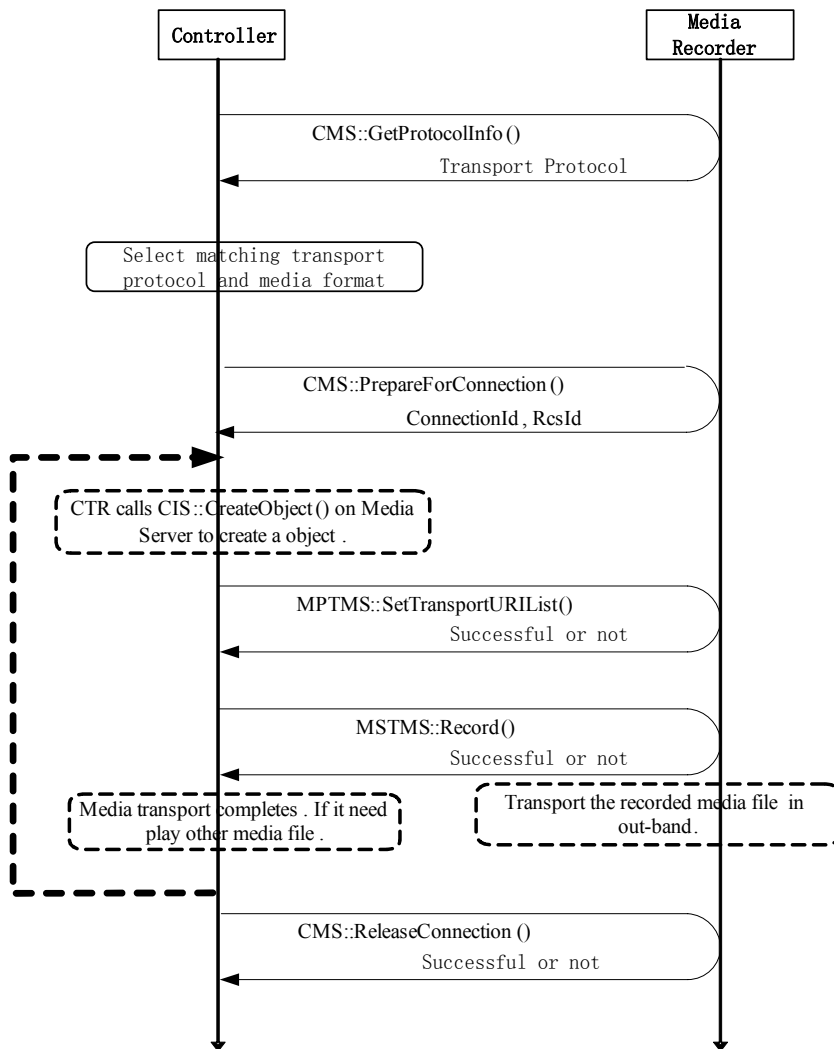
461 The device type definition of the MediaRecorder is as follows:

462 urn:IGRS:Device:DeviceType:MediaRecorder

463 **7.2.3.3 The Interaction Framework and Flow**

464 CTR controls the interaction between Media Recorder and Media Server to enable Media
465 Recorder to record specified media file and upload it to Media Server.

466 The interaction between Media Recorder and CTR is as follows (Figure 5):



467

468

Figure 5 - Interaction Flow between Media Recorder and CTR

469 The interaction flow between Media Recorder and CTR is as follows:

- 470 a) Through IGRS service discovery mechanism, CTR discovers the service on the
- 471 MediaRecorder device, including MP CMS, TMS and RMS;
- 472 b) Retrieve the transport protocol, control protocol and media format supported by MR: CTR
- 473 invokes the CMS::GetProtocolInfo() interface of CMS to get the transport protocol, control
- 474 protocol, media format supported by CMS. If multiple CMSs exist on the device, then CTR
- 475 should invoke CMS::GetProtocolInfo interfaces of each CMS to get transport protocol, control
- 476 protocol and media format supported by each of these CMSs.

- 477 c) Matching process managed by CTR: Match the transport protocol, control protocol and media
478 format supported by MS and other devices (MR).
- 479 d) Configure MR: Invoke CMS::PrepareForConnection() of CMS on MediaRecorder to notify MR
480 that a connection based on selected transport protocol and media format will be established.
481 The CMS of Recorder returns to CTR the connection ID. When the
482 CMS::PrepareForConnection() of CMS on MediaRecorder is invoked, with the available IP
483 address list returned by CMS::GetProtocolInfo() of MS as input, the output is the IP address
484 list that can be connected.
- 485 e) Select transport content: CTR invokes the MPTMS::SetTransportURLList() to identify the
486 content to be transported, which is located in the available IP address list from step d).
- 487 f) Record: CTR invokes MPTMS::Record() of Media Recorder to record the media;
- 488 g) Initiate media transport: Use MPTMS to upload the recorded media content to Media Server;
- 489 h) Release the connection: When the session between them is terminated, CTR shall invoke the
490 CMS::ReleaseConnection() of MediaRecorder to disconnect the connection.

491 **7.2.3.4 Sub-service and the Mandatory Interface Definitions**

492 This section lists the sub-services of Media Recorder (Table 7), including required and optional
493 services, and their interface list (Table 8).

494 **Table 7 – Sub-service List of Media Recorder**

Service Name	Required /Optional	Service Type	Field Explanation
Connection Management Service	Required	urn:IGRS:Service:ServiceType:ConnectionManagement:1	CMS is used to create and manage the connection between MS and MP. MS can support and manage several active connections at any one time by CMS. See section 8.2 in Application Profile – AV Profile for details.
Transport Management Service	Optional	urn:IGRS:Service:ServiceType:MediaPlayerTransportManagement:1	The optional MPTMS service enables CTR to adjust and control the transport of media stream on MP, such as play, pause, stop, seek etc. If this service exists, it means that the "Pull" Mode is supported. Otherwise, it is not supported. See section 8.3 in Application Profile – AV Profile for details.

495

496

Table 8 - Mandatory Interface List of Media Recorder

Service Name	Mandatory Interface	Field Explanation
Connection Management Service	GetProtocolInfo	Get the transport protocol supported by current device, including transport control protocol name, network port, content format, other protocol information and available IP address list on the MS.
	PrepareForConnection	Get the ConnectionId which is used to prepare and instantiate media data for sending and receiving, and TransportInstanceid which is used for media transport.
	ReleaseConnection	Notify device to disconnect the connection corresponding to ConnectionId and release resource.
	GetActiveConnectionIdList	Get current active connection ID list on MS or MP.
Media Player Transport Management Service	SetTransportURLList	Set TransportURI variable. Specify the URI list of media content object to be transport by MPTMS instance.
	GetTransportInfo	Get current transport state information of transport service instance.
	GetCurrentMediaInfo	Get media information transported by specified transport service instance.
	Record	Record in real-Time of the media stream on the specified media transport instance between MP and MS, and save the recording in a newly created media object on MS uploaded from MP.
	PauseRecord	Pause media content that is being recorded by MPTMS instance.
	ResumeRecord	Continue to record media content managed by MPTMS instance.
	StopRecord	Stop recording the media content by MPTMS instance.

497

498 7.2.4 File Server**499 7.2.4.1 Overview**

500 FileServer is the device that provides file content, and it includes FileAccessManagement
501 Service (FAMS) and FileConnectionManagement Service (FCMS).

502 The functions of FileServer include:

- 503 a) Based on the authentication information provided by FileClient, FileServer assigns
504 corresponding authentication key to FileClient (recommended validity only during the
505 session). This key is matched with FileClient access right such that the key can be used as
506 the identifier for subsequent file access by the FileClient;
- 507 b) Provide to the network with the file/directory that can be browsed or searched by FileClient;
- 508 c) Support simple management of file/directory in shared directory on FileServer.
- 509 d) Manage the uploading/downloading of file/directory;
- 510 e) Support FileServer file/directory update event subscription;
- 511 f) Support service update event subscription.

512 7.2.4.2 Device Type

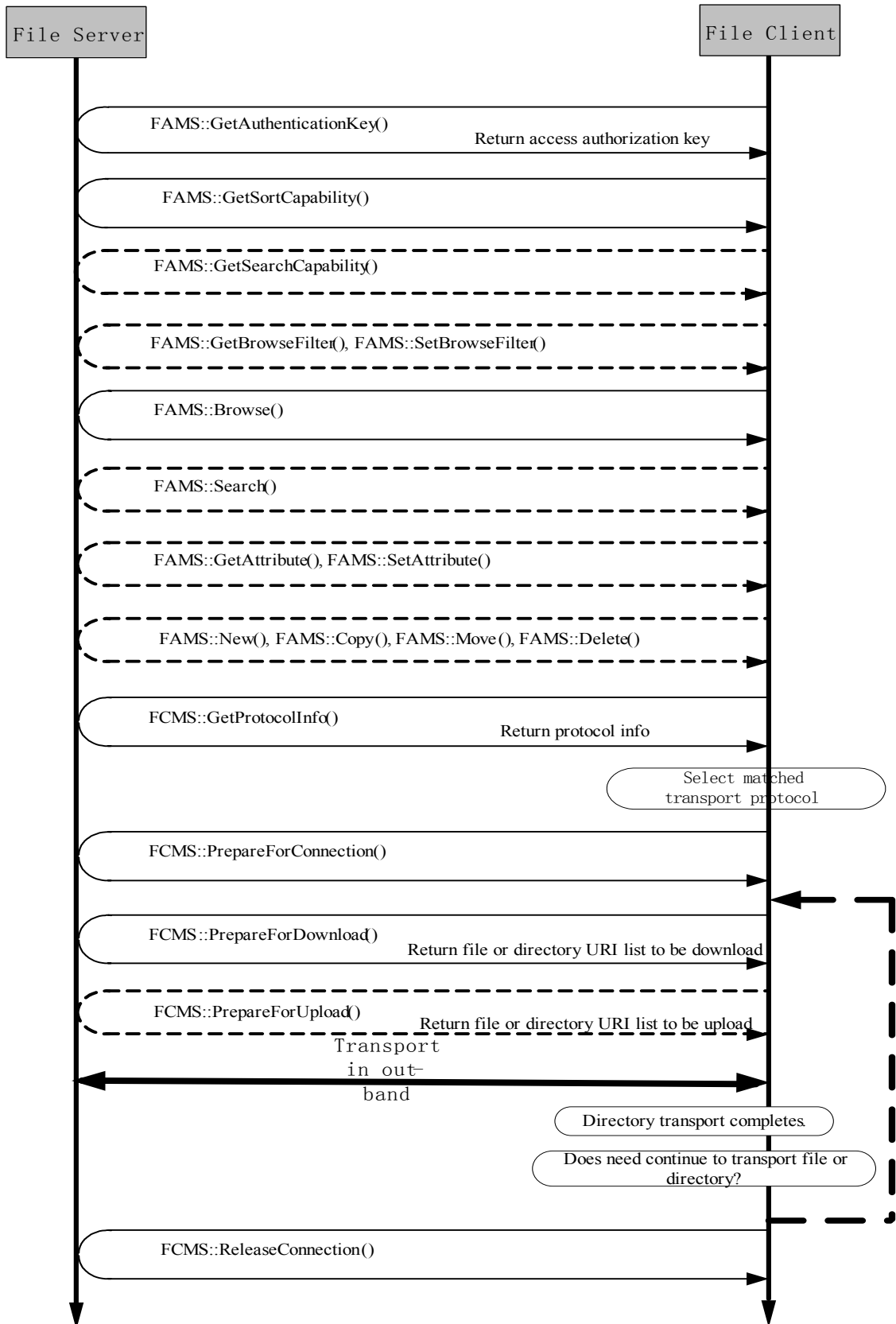
513 The device type definition of the FileServer is as follows:

514 urn:IGRS:Device:DeviceType:FileServer

515 7.2.4.3 The Interaction Framework and Flow

516 Through invoking the corresponding interfaces, FileServer and FileClient can implement
517 functions such as file upload/download etc.

518 The interaction framework between FileServer and FileClient is as follows (Figure 6):



519

520

Figure 6 - Interaction Flow between FileServer and FileClient

521

The interaction flow between FileServer and FileClient is as follows:

522 In complete interaction flow, the applicable interfaces include all required interfaces of
523 FileAccessManagement Service and FileConnectionManagement Service on FileServer and any
524 optional interfaces except those relate to subscription. FileClient can perform management
525 operations on shared files and directories, such as copy, move, delete, modify attributes, etc, or
526 upload file to FileServer, in addition to browse, search, download shared file and directory, or
527 retrieve attributes of shared file and directory.

528 In complete interaction flow, the invocation process of IGRS service interfaces between
529 FileClient and FileServer is described as follows:

- 530 a) IGRS service discovery: Through IGRS service discovery mechanism, FileClient discovers
531 FileAccessManagement Service and FileConnectionManagement Service on FileServer;
- 532 b) Retrieve access right: FileClient retrieves authentication key to get access right to shared file
533 on FileServer through invoking FAMS::GetAuthenticationKey() interface of
534 FileAccessManagement Service on FileServer. FileClient can use any combination
535 information of DeviceID/DeviceName, UserName/Password, and Third-party authentication
536 mode of the local device, to retrieve authentication key from FileServer;
- 537 c) Browse and search shared file/directory on FileServer:
 - 538 1) Retrieve sorting and searching capability of FileServer: FileClient can retrieve shared
539 file/directory sorting capability of FileServer through invoking FAMS::GetSortCapability()
540 interface of FileAccessManagement Service on FileServer. For example, it can sort
541 according to file names or time of modification. It can also retrieve shared file/directory
542 searching capability of FileServer through invoking FAMS::GetSearchCapability ()
543 interface of FileAccessManagement Service on FileServer. For example, it can search
544 according to file names or time of modification;
 - 545 2) Retrieve or set browsing filter of shared file/directory: FileClient can retrieve or set
546 browsing file of shared file/directory through invoking FAMS::GetBrowseFilter() or
547 FAMS::SetBrowseFilter() interface of FileAccessManagement Service on FileServer;
 - 548 3) Browse or search shared file/directory on FileServer: FileClient can browse all files and
549 sub-directory information in any specified shared directory through invoking
550 FAMS::Browse() interface of FileAccessManagement Service on FileServer; It can also
551 search shared file/directory according to designated criteria through invoking
552 FAMS::Search() interface of FileAccessManagement Service on FileServer;
 - 553 4) Retrieve or set attributes of shared file/directory: FileClient can retrieve attributes of
554 specified shared file/directory through invoking FAMS::GetAttribute() interface of
555 FileAccessManagement Service on FileServer. It can also set attributes of specified
556 shared file/directory through invoking FAMS::SetAttribute() interface of
557 FileAccessManagement Service on FileServer;
- 558 d) Manage shared file/directory: FileClient can create new shared file/directory in specified
559 shared directory on FileServer through invoking FAMS::New() interface of
560 FileAccessManagement Service on FileServer; or delete shared file/directory on FileServer,
561 through invoking FAMS::Delete() interface of FileAccessManagement Service; or copy a
562 shared file/directory on FileServer to a specified shared directory through invoking
563 FAMS::Copy() interface of FileAccessManagement Service; or move a shared file/directory to
564 specified shared directory through invoking FAMS::Move() interface of
565 FileAccessManagement Service;
- 566 e) Setup Connection with FileServer
 - 567 1) Retrieve transport protocols supported by FileServer: FileClient retrieves transport
568 protocols supported by FileServer, through invoking FCMS::GetProtocollInfo() interface of
569 FileConnectionManagement Service on FileServer;
 - 570 2) Select matching transport protocols: Using retrieved transport protocols supported by
571 FileServer according to 1) in step e), FileClient selects matching transport protocols
572 supported by the device;
 - 573 3) Connection preparation: FileClient notifies FileServer to prepare for connection setup, and
574 retrieve connection identifier for connection management to use in subsequent interaction

575 process through invoking FCMS::PrepareforConnection() interface of
 576 FileConnectionManagement Service in FileServer;

577 f) Download shared file/directory from FileServer or upload shared file/directory to FileServer:
 578 FileClient can setup the downloading of shared file/directory, and retrieve the URI list of
 579 shared file/directory to be downloaded through invoking FAMS::PrepareforDownload()
 580 interface of FileAccessManagement Service on FileServer; or setup file/directory uploading of
 581 shared directory, and retrieve the URI of the shared directory through invoking
 582 FAMS::PrepareforUpload() interface of FileAccessManagement Service;

583 g) File transport: FileClient and FileServer use out-of-band transport protocols and the URI
 584 retrieved in step f) to download shared files from FileServer or upload client file to the
 585 designated shared directory on FileServer. After the transporting of the specified file is
 586 completed, it returns to step f) if required;

587 h) Close connection and release resource: When the file transport has been completed,
 588 FileClient notifies FileServer to close the connection between them and release the resource
 589 through invoking FCMS::ReleaseConnection() interface of FileConnectionManagement
 590 Service on FileServer.

591 **7.2.4.4 Sub-service and the Mandatory Interface Definitions**

592 This section lists the sub-services of File Server (Table 9), including required and optional
 593 services, and their interface list (Table 10).

594 **Table 9 – Sub-service List of File Server**

Service Name	Required /Optional	Service Type	Field Explanation
File Access Management Service	Required	urn:IGRS:Service:Service Type:FileAccessManagement:1	FileAccessManagement Service provides the following functions to FileClient: a) Provide authentication for FileClient, and thus give corresponding file access right to FileClient; b) Allow FileClient to retrieve sorting/searching capability supported by FileServer; c) Allow FileClient to browse content directory in the network provided by FileServer; d) Allow FileClient to search specified file/directory; e) Allow FileClient to retrieve and modify attributes of file/directory; f) Allow FileClient to retrieve and set browsing filter; g) Allow FileClient to subscribe to file/directory object update event; h) Allow FileClient to subscribe to FileAccessManagement Service update event; i) Support FileClient to upload and download the specified file/directory. See section 7.2 in Application Profile – File Profile for details.
File Connection Management Service	requisite	urn:IGRS:Service:Service Type:FileConnectionManagement:1	FileConnectionManagement Service is used to create and manage the connection between FileServer and FileClient. FileServer can support and manage multiple active connections at any one time through FileConnectionManagement Service. See section 7.3 in Application Profile – File Profile

Service Name	Required /Optional	Service Type	Field Explanation
			for details.

595

596

Table 10 - Mandatory Interface List of File Server

Service Name	Mandatory Interface	Field Explanation
File Access Management Service	GetAuthenticationKey	Retrieve assigned authentication key of FileServer based on the authentication information. The authentication key specifies the access rights of FileClient. The authentication key is recommended to be only in effect during a session. During the subsequent service invocations by FileClient, the authentication key is required to be used as the input parameter.
	GetSortCapability	Retrieve file/directory sorting capability supported by FileAccessManagement Service on FileServer such that FileClient can use the file/directory attribute set as the sorting rule, such as name, time of creation, etc.
	Browse	Return to the lower level file/directory list of the specified directory on FileServer, and get back the basic attributes of each object.
	GetAttribute	Retrieve attribute set of the specified file/directory object on FileServer.
	PrepareforDownload	Prepare for file download, and pre-detect the feasibility of the file/ directory download. If download is allowed by FileServer, then the URI of the source to be transported shall be returned to the client, so that the FileClient application can download the file/directory through out-of-band data transport; or else corresponding error codes shall be returned.
File Connection Management Service	GetProtocolInfo	Retrieve the transport protocol supported by FileServer, including the transport control protocol name, network port, the content format and other protocol information, as well as the valid IP address list of FileServer.
	PrepareforConnection	Retrieve the ConnectionId used to send and receive data.
	ReleaseConnection	Notify the device to close the connection corresponding to the ConnectionId, and release resource.
	GetActiveConnectionIdList	Retrieve the current active connection identifier list on FileServer.

597

Bibliography

- 598 ISO/IEC 14543-5-2, *Information Technology - Information Device Intelligent Grouping and*
599 *Resource Sharing Specification Part 2: Application Profile*
- 600 ISO/IEC 14543-5-6, *Information Technology - Information Device Intelligent Grouping and*
601 *Resource Sharing Specification Part 6: Service Type*