Car Connectivity Consortium

MirrorLink®

Audio

Version 1.1.3
(CCC-TS-012)

VERSION HISTORY

Version	Date	Comment
1.0	06 October 2010	Public release
1.0.1	26 June 2011	Approved Version
1.1	31 March 2012	Approved Version
1.1.1	24 September 2012	Approved Errata Version
1.1.2	05 March 2013	Approved Errata Version
1.1.3	21 November 2013	Approved Errata Version

LIST OF CONTRIBUTORS

Beckmann, Mark	Volkswagen AG
Benesch, Matthias	Daimler AG
Bose, Raja,	Nokia Corporation
Brakensiek, Jörg (Editor)	Nokia Corporation
Fernahl, Dennis	Carmeq (for Volkswagen AG)
Kostiainen, Kari	Nokia Corporation
Lehner, Martin	Jambit
Park, Keun-Young	Nokia Corporation
Park, Tommy	LG Electronics
Wolf, Michael	Daimler AG
Yoon, Young-Rang	LG Electronics

LEGAL NOTICE

TABLE OF CONTENTS

VERSION HISTORY **2**
LIST OF CONTRIBUTORS **2**
LEGAL NOTICE **3**
TABLE OF CONTENTS **4**
LIST OF FIGURES **5**
LIST OF TABLES **6**
TERMS AND ABBREVIATIONS **7**
1 ABOUT **8**
2 INTRODUCTION **9**
3 AUDIO LINK OPTIONS **10**
4 AUDIO LINK SELECTION **13**
4.1 IDENTIFICATION OF AVAILABLE AUDIO LINKS 13
4.2 LAUNCHAPPLICATION (APPID, PROFILEID) 13
4.3 TERMINATEAPPLICATION (APPID, PROFILEID) 14
4.4 GETAPPLICATIONSTATUS (APPID, PROFILEID) 15
5 RTP AUDIO STREAMING **16**
5.1 RTP PACKET STRUCTURE AND HEADER DEFINITION 16
5.2 RTP AUDIO PAYLOAD DEFINITION 18
5.2.1 16 Bit Audio Payload (Mono) *18*
5.2.2 16 Bit Audio Payload (Stereo) *19*
5.3 ESTABLISHING THE RTP CONNECTION 19
5.3.1 RTP Server within MirrorLink Server *19*
5.3.2 RTP Server within MirrorLink Client *20*
5.4 CLIENT AND SERVER IMPLEMENTATION 20
6 VOICE COMMAND HANDLING **23**
6.1 VOICE COMMAND OVER RTP WITH ESTABLISHED VNC SESSION 23
7 INTEROPERABILITY WITH BLUETOOTH **25**
7.1 BLUETOOTH PROFILES RELEVANT FOR MIRRORLINK 25
7.2 INTEROPERABILITY STATES –MIRRORLINK SERVER PERSPECTIVE 25
7.3 INTEROPERABILITY STATES –MIRRORLINK CLIENT PERSPECTIVE 27
8 AUDIO SIGNAL PROCESSING CONFIGURATION **30**
8.1 CONVERSATIONAL AND TELEPHONY-BASED AUDIO 30
8.2 BLUETOOTH HFP NOISE REDUCTION/ECHO CANCELLATION 30
8.3 RTP-BASED CONVERSATIONAL AUDIO 30
9 REFERENCES **31**

LIST OF FIGURES

Figure 1: Audio Setup 9
Figure 3: Message Flow – Terminate Audio Link 15
Figure 4: Sequence to Establish the RTP Connection 20
Figure 5: Voice Command Session over RTP 23
Figure 6: Voice Command Dialog over RTP 24
Figure 7: Voice Command Session over RTP with ongoing Phone Call 24
Figure 8: Bluetooth / MirrorLink IOP States – MirrorLink Server Perspective 26
Figure 9: Bluetooth / MirrorLink IOP States – MirrorLink Client Perspective 28

LIST OF TABLES

Table 1: UPnP Negotiation for Audio Selection 11
Table 2: UPnP Negotiation for Audio Selection 12
Table 3: RTP Packet Header Definition 16
Table 4: RTP Packet Header Extension 17
Table 5: RTP Payload Format – 16 Bit Mono 19
Table 6: RTP Payload Format – 16 Bit Stereo 19
Table 7: IOP Transition (from MirrorLink Server perspective) 27
Table 8: IOP Transition (from MirrorLink Client perspective) 29

TERMS AND ABBREVIATIONS

A2DP	Bluetooth Advanced Audio Distribution Profile
ARP	Address Resolution Protocol
BT	Bluetooth
CDC	Communications Device Class; specified from USB Device Working Group
CE	Consumer Electronics; CE devices are referred to as mobile devices within this specification
DHCP	Dynamic Host Configuration Protocol
HFP	Bluetooth Hands-free Profile
HSP	Bluetooth Headset Profile
HMI	Human Machine Interface
HU	Head-unit (this term is used interchangeably with the MirrorLink client)
HS	Head-set
IP	Internet Protocol
NCM	Network Control Model; part of the CDC device class
Pointer Event	Pointer events are used to describe touch screen action in which the user touches the screen with one (virtual) finger only at a single location.
RFB	Remote Framebuffer
RTP	Real-time Transport Protocol
TCP	Transmission Control Protocol
Touch Event	Touch events are used to describe touch screen action in which the user touches the screen with two or more separate fingers at different locations. Touch events are used to describe more complex touch action, like pinch-open or pinch-close.
UDP	User Datagram Protocol
UI	User Interface
UPnP	Universal Plug and Play
USB	Universal Serial Bus
VNC	Virtual Network Computing

1 ABOUT

This document is part of the MirrorLink specification, which specifies an interface for enabling remote user interaction of a mobile device via another device. This specification is written having a vehicle head-unit to interact with the mobile device in mind, but it will similarly apply for other devices, which provide a color display, audio input/output and user input mechanisms.

The document will focus on the interface functionality, its parameters and protocols only. It does not provide any guidelines for implementing the protocol. If there is a reference towards an implementation, this is of an informative nature only.

The specification lists a series of requirements, either explicitly or within the text, which are mandatory elements for compliant solutions. Recommendations are given, to ensure optimal usage and to provide suitable performance. All recommendations are optional.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document follow the notation as described in RFC 2119 [7].

1. MUST: This word, or the terms "REQUIRED" or "SHALL", means that the definition is an absolute requirement of the specification.
2. MUST NOT: This phrase, or the phrase "SHALL NOT", means that the definition is an absolute prohibition of the specification.
3. SHOULD: This word, or the adjective "RECOMMENDED", means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
4. SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED" means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
5. MAY: This word, or the adjective "OPTIONAL", means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

2 INTRODUCTION

This specification defines how the MirrorLink client selects the transport media that the MirrorLink server MUST use to route audio streams. The MirrorLink server distinguishes between two audio streams, namely phone and application audio. It advertises available transport options, using UPnP TmServerDevice:1 services. Audio streams are specified for the following remote access protocols:

- RTP Real-Time Transport Protocol
- BTA2DP Bluetooth Advanced Audio Distribution Profile
- BTHFP Bluetooth Hands Free Profile

It is the MirrorLink client's responsibility to select from the advertised audio transport mechanisms how the MirrorLink server MUST stream the different audio sources. The audio link selection is done according the following priorities (highest priority first):

1. Keep existing Bluetooth HFP or A2DP connection to another external device, which is not a MirrorLink client, if overriding the resource assignment is not allowed.
2. Follow audio link selection using the mechanism described in this section.
3. Manual Bluetooth pairing (same behavior as in non-MirrorLink use cases)
4. Phone speaker & microphone (same behavior as in non-MirrorLink and non-Bluetooth use cases)

The specification allows for different transport mechanisms based on the selections the MirrorLink client has taken.

The MirrorLink audio architecture, as shown in Figure 1, allows using the Real-time Transport Protocol for streaming audio captured from the mobile device, to the MirrorLink client. The audio output from the mobile device is streamed in an application agnostic manner so that it does not require re-design or modification of existing applications running on the device.

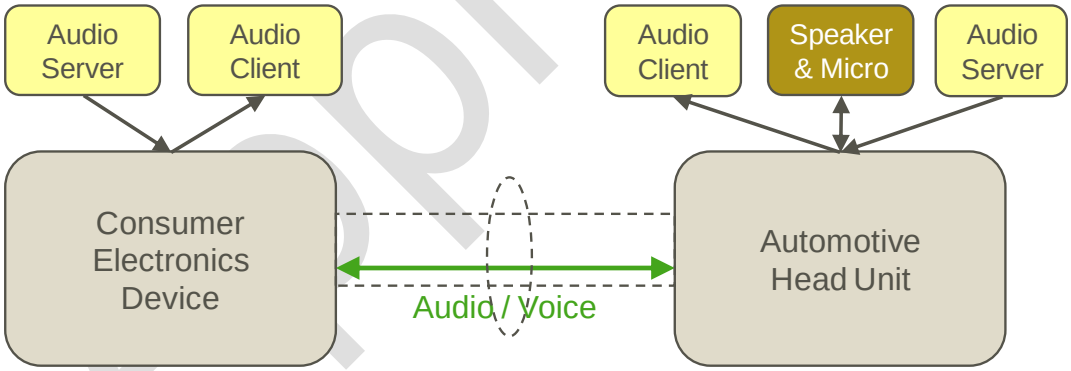


Figure 1: Audio Setup

This specification covers both audio output from and audio input to the MirrorLink Server device. Unless otherwise stated, the specification applies for the audio server and the audio client in the same way.

- The audio output will be handled from the audio server on the MirrorLink server and the audio client on the MirrorLink client.
- The audio input will be handled from the audio client on the MirrorLink server and the audio server on the MirrorLink client.

In addition to RTP, the MirrorLink specification allows regular Bluetooth audio connectivity for both phone and media audio streams.

3 AUDIO LINK OPTIONS

MirrorLink allows the MirrorLink Client and Server to use the following audio features:

- Media – Streaming of media audio from the MirrorLink Server to the MirrorLink Client
- Phone – Bidirectional streaming of conversational audio
- VC – Streaming of voice command (VC) audio from the MirrorLink Client to the MirrorLink Server

The above listed audio features are available over the following connectivity options:

- Bluetooth (BT)
- RTP

The MirrorLink Server MUST advertise the audio features and connectivity options it supports within the UPnP A_ARG_TYPE_AppList application listings. The MirrorLink Server MUST advertise individual audio components, with all combinations of the supported audio content categories[1]. This will allow the MirrorLink Client to specifically inform the MirrorLink Server, which feature it is going to use. The possible audio content categories for the different audio components are listed below:

- RTP Client: Phone Audio, Voice Command In, Phone Audio + Voice Command In
- RTP Server: Media Audio Out, Phone Audio + Media Audio Out
- BT A2DP: Media Audio Out
- BT HFP: Phone Audio, Phone Audio + Voice Command In

The MirrorLink Server MUST include the RTP Server with a value of `audioInfo@contentCategory` equaling `Media Audio Out (0x01)` first in the list of advertised RTP Servers for the same RTP payload types (A_ARG_TYPE_AppList).

If the MirrorLink Server supports Voice Command over RTP, it MUST include the RTP Client with a value of `audioInfo@contentCategory` equaling `Voice Command In (0x10)` first in the list of advertised RTP Clients for the same RTP payload types (A_ARG_TYPE_AppList).

The MirrorLink Client MUST select the audio features and their connectivity option, which the MirrorLink Client is going to use. The MirrorLink Client MUST use the UPnP Application Server service's Launch Application action to inform the MirrorLink Sever about its selection. The MirrorLink Client MUST NOT launch more than one RTP Client, RTP Server, BT HFP, and BT A2DP component.

Based on the set of audio components, launched from the MirrorLink Client, the MirrorLink Server and Client MUST use specific audio links for Media, Phone and Voice Command use cases. Table 1 lists the possible combinations of audio features and their underlying connectivity options on the left side. For each combination, the required audio components are listed, which MUST be launched from the MirrorLink Client to enable it. Note, that some Audio Feature Set combinations are not possible.

Audio Feature Combinations and underlying Connectivity			Audio Components launched from the MirrorLink Client to enable the Audio Feature Combination			
Media	**Phone**	**VC**	**BT HFP**	**BT A2DP**	**RTP Server**	**RTP Client**
-	-	-	-	-	-	-
-	-	RTP	-	-	-	VC
-	-	BT	Not possible			
-	RTP	-	-	-	Phone	Phone
-	RTP	RTP	-	-	Phone	Phone + VC

[1] Example: If the MirrorLink supports an RTP Server for Media and Phone Audio, it MUST advertise three RTP Servers, one showing Media, one Phone, and one Media & Phone support.

Audio Feature Combinations and underlying Connectivity			Audio Components launched from the MirrorLink Client to enable the Audio Feature Combination			
Media	**Phone**	**VC**	**BT HFP**	**BT A2DP**	**RTP Server**	**RTP Client**
-	RTP	BT	Not possible			
-	BT	-	Phone	-	-	-
-	BT	RTP	Phone	-	-	VC
-	BT	BT	Phone + VC	-	-	-
RTP	-	-	-	-	Media	-
RTP	-	RTP	-	-	Media	VC
RTP	-	BT	Not possible			
RTP	RTP	-	-	-	Phone + Media	Phone
RTP	RTP	RTP	-	-	Phone + Media	Phone + VC
RTP	RTP	BT	Not possible			
RTP	BT	-	Phone	-	Media	-
RTP	BT	RTP	Phone	-	Media	VC
RTP	BT	BT	Phone + VC	-	Media	-
BT	-	-	-	Media	-	-
BT	-	RTP	-	Media	-	VC
BT	-	BT	Not possible			
BT	RTP	-	-	Media	Phone	Phone
BT	RTP	RTP	-	Media	Phone	Phone + VC
BT	RTP	BT	Not possible			
BT	BT	-	Phone	Media	-	-
BT	BT	RTP	Phone	Media	-	VC
BT	BT	BT	Phone + VC	Media	-	-

Table 1: UPnP Negotiation for Audio Selection

BT HFP and BT A2DP MAY be connected outside the MirrorLink session, i.e. without specifically using the UPnP mechanisms to launch those. In that case the MirrorLink Client and Server MUST make the selection based on the following table. Components marked with `Legacy` MUST NOT be launched using the UPnP Application Sever service's Launch Application action. The setup of the legacy Bluetooth connection is outside the scope of the MirrorLink specifications.

Audio Feature Set and underlying Connectivity			Audio Components launched from the MirrorLink Client to enable the Audio Feature Set			
Media	**Phone**	**VC**	**BT HFP**	**BT A2DP**	**RTP Server**	**RTP Client**
-	BT	-	Legacy	-	-	-
-	BT	RTP	Legacy	-	-	VC
-	BT	BT	Legacy	-	-	-
RTP	BT	-	Legacy	-	Media	-
RTP	BT	RTP	Legacy	-	Media	VC
RTP	BT	BT	Legacy	-	Media	-
BT	-	-	-	Legacy	-	-
BT	-	RTP	-	Legacy	-	VC

Audio Feature Set and underlying Connectivity			Audio Components launched from the MirrorLink Client to enable the Audio Feature Set			
Media	**Phone**	**VC**	**BT HFP**	**BT A2DP**	**RTP Server**	**RTP Client**
BT	-	BT	Not possible			
BT	RTP	-	-	Legacy	Phone	Phone
BT	RTP	RTP	-	Legacy	Phone	Phone + VC
BT	RTP	BT	Not possible			
BT	BT	-	Legacy	Legacy	-	-
BT	BT	RTP	Legacy	Legacy	-	VC
BT	BT	BT	Legacy	Legacy	-	-

Table 2: UPnP Negotiation for Audio Selection with Legacy Bluetooth

If an audio use feature is not covered from the audio component selection, done by the MirrorLink Client, the MirrorLink Server MUST fallback to its default configuration, which MAY include using its integrated microphone and speakers.

4 AUDIO LINK SELECTION

The TmApplicationServer:1's service MUST be used for controlling audio streams. Each type of audio source or sink on the MirrorLink server is considered a remote application which can be remotely controlled by the MirrorLink Control Point using TmApplicationServer:1 service.

The audio servers and clients can be started and terminated the same way as any other remote application using the LaunchApplication() and TerminateApplication() SOAP actions respectively. The audio server, optionally running as part of the MirrorLink client, will provide audio input like voice control to the MirrorLink server.

Next a description is provided of how TmApplicationServer:1's SOAP actions are utilized to select the audio links. Note that only the aspects specific to audio link selection are covered here and the reader is REQUIRED to refer to the corresponding service specification for complete details of the TmApplicationServer:1.

The MirrorLink client SHOULD make the audio link selection not later than 10s after receiving the first A_ARG_TYPE_AppList response from the MirrorLink server.

The MirrorLink client MUST have made the audio link selection prior starting the first VNC based remote application.

4.1 Identification of Available Audio Links

The identification of audio links is described in [13].

4.2 LaunchApplication (AppID, ProfileID)

The LaunchApplication() action MUST be used to start the audio streaming on the MirrorLink server side and therefore select the underlying audio link. The response received will be an URI to the audio streaming sources/sinks using the audio streaming protocol identifier. The URI MUST follow the A_ARG_TYPE_URI definition as specified in [13].

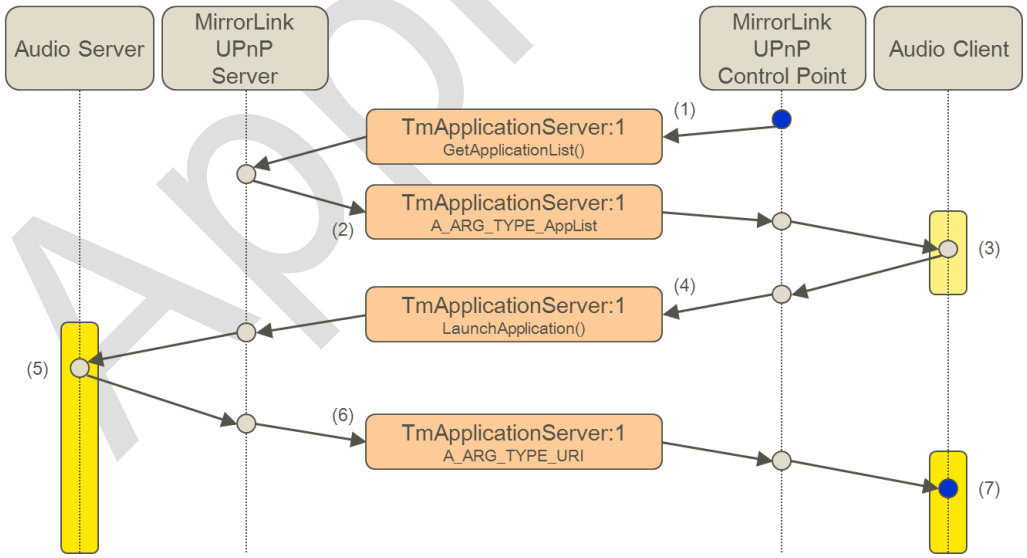


Figure 2: Message Flow – Launch Audio Link

The message flow for selecting an audio link, using LaunchApplication(), as shown in Figure 2, consists of the following steps:

1. MirrorLink UPnP Control Point: Call GetApplicationList() SOAP action

2. MirrorLink UPnP Server: Return A_ARG_TYPE_AppList, which includes a list of available audio servers
3. MirrorLink Client: Select the preferred audio server, using the protocolID and direction elements
    a. BT only: Prepare for BT connection, if MirrorLink server is expected to initiate the BT connection[2]
4. MirrorLink UPnP Control Point: Call LaunchApplication() SOAP action containing respective audio server application ID
5. MirrorLink Server: Start selected audio server
    a. Determine new audio link configuration, using Table 1.
    b. RTP only: Prepare RTP streaming[3]
    c. BT only: Prepare for BT connection.
    d. BT only: Initiate BT connection if MirrorLink server is expected to initiate the BT connection.
6. MirrorLink UPnP Server: Return A_ARG_TYPE_URI representing the audio server URI
7. MirrorLink Client: Start audio client
    a. RTP only: Connect to RTP streaming port
    b. BT only: Initiate BT connection, if MirrorLink client is expected to initiate the BT connection.

Using LaunchApplication() will enable a specific audio link. This link is valid until TerminateApplication() action with the same AppID is called, or the MirrorLink session is closed.

If Bluetooth is already turned on, the MirrorLink server MUST accept BT connection requests after responding to launching a specific Bluetooth profile. If Bluetooth is turned off, the MirrorLink server MAY switch on Bluetooth before responding to launching a specific Bluetooth profile.

The MirrorLink server MUST only initiate a Bluetooth connection, if the MirrorLink client has specifically requested the server to do so, setting <startConnection> in A_ARG_TYPE_ClientProfile to “false”. To ensure automatic pairing, without user intervention, the MirrorLink client SHOULD provide its Bluetooth MAC address (bdAddr) in A_ARG_TYPE_ClientProfile.

If the MirrorLink server does not support Bluetooth connection initialization, i.e. it has specifically set <startConnection> in the UPnP TmServerDevice:1 device XML to “false”, and the MirrorLink client does not support Bluetooth connection initialization either, the MirrorLink client MUST NOT use the LaunchApplication() to start a Bluetooth connection.

4.3 TerminateApplication (AppID, ProfileID)

The TerminateApplication() action MUST be used to stop the audio streaming (in either direction) on the MirrorLink server side and follows the specification [13]. Invoking the TerminateApplication action causes the corresponding audio link to be closed.

[2] The MirrorLink client can indicate to the server via the SetClientProfile action that the server MUST initiate the BT connection. By default, the MirrorLink server will not start the BT connection.

[3] RTP streaming is done over UDP.

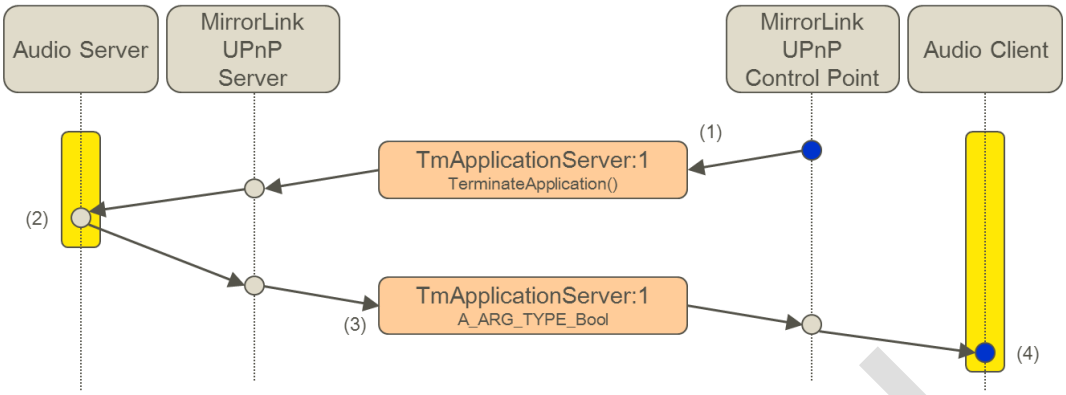


Figure 3: Message Flow – Terminate Audio Link

The message flow for unselecting an audio link, using TerminateApplication(), as shown in Figure 3, consists of the following steps:

1. MirrorLink UPnP Control Point: Call TerminateApplication() SOAP action containing respective audio server application ID
2. MirrorLink Server: Stop audio server
    a. Determine new audio link configuration, using Table 1
    b. RTP only: Stop the RTP streaming
    c. BT only: Disconnect BT profile; optionally power down BT
3. MirrorLink UPnP Server: Return termination success response (true or false)
4. MirrorLink Client: Stop audio client
    a. RTP only: Disconnect from RTP streaming port
    b. BT only: Disconnect BT profile; optionally power down BT

In case the TerminateApplication() action is used to terminate the audio server residing in the MirrorLink server device then the MirrorLink client will stop receiving the audio stream from the MirrorLink server.

In case the TerminateApplication() action is used to terminate the audio client residing in the MirrorLink server then the MirrorLink server will stop receiving the audio stream from the MirrorLink client.

4.4 GetApplicationStatus (AppID, ProfileID)

The GetApplicationStatus() action provides the current status of an audio server or client running on the MirrorLink server and is following [13].

The return values (of the type A_ARG_TYPE_AppStatus) specified for this SOAP action can be any of the following:

- Foreground: Audio link is launched.
- Background: Not used.
- Notrunning: Audio link is terminated / not launched.

5 RTP AUDIO STREAMING

The audio RTP server and client on the MirrorLink server listen on pre-specified ports, which are advertised using UPnP mechanisms.

When the audio server captures audio data, it will encode the audio into RTP packets using the negotiated RTP Payload type and transmit the RTP packets over UDP/IP.

The audio client is fully responsible for receiving and decoding the data packets and restoring the packet order if they arrive in out of order sequence. More detailed information about the RTP packet structure can be found later in the document.

The MirrorLink server MUST support RTP audio streaming for unidirectional audio to the MirrorLink client. The MirrorLink Client MUST support RTP audio streaming for unidirectional audio from the MirrorLink Server. The MirrorLink server MAY support RTP audio streaming for bi-directional audio.[4]

5.1 RTP Packet Structure and Header Definition

RTP packets contain the standard RTP message header and the payload. Usually each RTP packet audio payload contains a predefined amount of audio data, but in a special case of end of stream (M=1), payload can be zero length. Therefore, the RTP client SHOULD NOT assume fixed payload length. Each RTP packet audio payload contains predefined amounts of audio data. Audio samples are sent in sequential order (in sampling order, first sample first).

Each RTP packet contains the standard header as defined in IETF RFC3550 [1]. The header fields and their default values are described in the following section. The RTP packet structure is shown in Table 3.

00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
V		P	X	CC				M	PT							Sequence Number															
Timestamp																															
SSRC																															
CSRC [0..15]																															

Table 3: RTP Packet Header Definition[5]

The first twelve octets are present in every RTP packet, while the list of CSRC identifiers is present only when inserted by a mixer. The following fields are the RTP specification, unless otherwise mentioned:

- **Version (V) : 2 bits**
  The RTP version defined by this specification is two (2).
- **Padding (P) : 1 bit**
  If the padding bit is set, the packet contains one or more additional padding octets at the end which are not part of the payload.
- **Extension (X) : 1 bit**

[4] If bi-directional RTP streaming is not supported, telephony use cases work only if BT HFP is supported.

[5] Note: The RTP header definition keeps the IETF format for numbering bits. Therefore the most significant bit is 0 for all RTP header and payload descriptions.

If the extension bit is set, the fixed header MUST be followed by exactly one header extension. If the RTP header carries information about the audio category and application id, then this bit MUST be one (1).

- **CSRC count (CC) : 4 bits**
  The CSRC count contains the number of CSRC identifiers that follow the fixed header.
- **Marker (M) : 1 bit**
  The interpretation of the marker is defined (in reference to RFC 2190 [8]);

  0: More packets will follow.

  1: Current package carries the end of stream[6]
- **Payload type (PT) : 7 bits**
  This field identifies the format of the RTP payload and determines its interpretation by the application.
- **Sequence number : 16 bits**
  The sequence number increments by one for each RTP data packet sent, and MAY be used by the receiver to detect packet loss and to restore packet sequence.
- **Timestamp : 32 bits**
  The timestamp reflects the sampling instant of the first octet in the RTP data packet. The initial value of the timestamp is random.
- **SSRC Synchronization source : 32 bits**
  This field identifies the synchronization source.
- **CSRC Contributing Source : 32 bits**
  An array of 0 to 15 CSRC elements identifying the contributing sources for the payload contained in this packet. The number of identifiers is given by the *CC* field.

If the RTP header carries information about the audio sources, the RTP extension header MUST be used as shown in Table 4. The RTP header extension MUST follow section 5.3.1 of RFC 3550 [1].

00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Profile Identifier																Length															
Header Extension::Application Identifier																															
Header Extension::Application Category																															

Table 4: RTP Packet Header Extension

- **Profile Identifier: 16 bits**
  Profile identifier for the extension header; MUST be 0x388C.
- **Length: 16 bits**
  Number of 32-bit words for the extension, excluding the extension header (therefore 0 is a valid length). Length MUST be an even number.
- **Header Extension: Array of 64 bits**
  In case of a length of greater than zero, the Header Extension consists of one or more entries of 64 bits each. Each 64 bit entry references one application contributing to the audio content within the RTP payload. The entry with the highest priority MUST be given first. Other entries MUST follow

[6] Audio client may use this information to e.g. empty any available receive buffer.

in decreasing priority order. Priority order is defined by the weights of the audio entries within the audio mixing stage.

The MirrorLink Server SHOULD only list significant audio sources.

- The first 32 bits give the unique application id. For an application being advertised via UPnP, the unique application id MUST match the advertised appID. This field MAY be left empty (i.e. zero value).
- The second 32 bits give the application category, as defined in [13].

The audio server MUST use RTP Packet Header Extensions to provide the application category and the application identifier of the transmitted audio stream.

The MirrorLink Server MUST use the audio context information, provided from MirrorLink certified applications via the Common API. The MirrorLink Server SHOULD determine the source of audio streams originating from non-certified MirrorLink applications and from applications not using the Common API [16]. The MirrorLink Server MUST set the audio context information to valid, not unknown values, if it can determine the source of an audio stream. The MirrorLink Server MUST set the audio context information to 0x00000000, if it cannot determine the source of an audio stream.

RTP packets without a valid RTP Header Extension SHOULD be treated similar to RTP packets received from an unknown application.

5.2 RTP Audio Payload Definition

RTP payload length and sampling frequency MUST be negotiated beforehand. This MUST be done using UPnP mechanisms as defined in [13] and [14]. The following paragraphs define the audio payload format for 8 and 16 bit audio samples.

The RTP server SHOULD support payload type 0 (8 bit, 8 kHz, mono)[7]. The RTP server MUST support payload type 99 (16 bit, stereo, 48 kHz). The RTP client MUST support payload type 99. The audio server and client MAY support other standardized RTP payload types as well. The RTP server MUST use a payload type, supported from the RTP client.

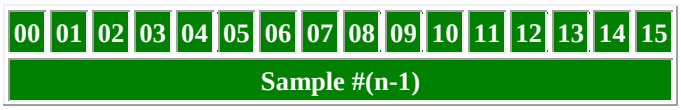
5.2.1 16 Bit Audio Payload (Mono)

This payload type denotes uncompressed audio data samples using 16-bit signed representation with 65,535 equally divided steps between minimum and maximum signal levels, ranging from -32,768 to 32,767. The value is represented in two's complement notation and transmitted in network byte order (most significant byte first).

The audio data has the following properties:

- One audio channels (mono)
- 16 bits
- Frequency: 48 kHz
- Payload type: 98

Each audio sample is stored to the RTP payload using 16 bits. Each sample is stored to the payload using the order it was taken (first sample first).



[7] Audio stream is mu-law encoded [11].

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
Sample #n															

Table 5: RTP Payload Format – 16 Bit Mono

Note: Except for the sampling frequency, this payload type is identical to payload type 11 (L16, mono, 44.1 KHz), as defined in RFC 3551 [11].

5.2.2 16 Bit Audio Payload (Stereo)

This payload type denotes uncompressed audio data samples using 16-bit signed representation with 65,535 equally divided steps between minimum and maximum signal level, ranging from -32,768 to 32,767. The value is represented in two's complement notation and transmitted in network byte order (most significant byte first).

The audio data has the following properties:

- Two audio channels (stereo).
- 32 bits (16 bits per channel).
- Frequency: 48 kHz
- Audio data for each channel interleaved.
- Payload type: 99

Each audio sample is stored to the RTP payload using 32 bits. Each sample is stored to the payload using the order it was taken (first sample first). Audio sample's left channel data (16 bits) is stored first and then the right channel data (16 bits). This process is applied for each of the audio samples. Audio payload always contains both the right and left channel data, as shown in Table 6. Channel data is never divided amongst different RTP packets.

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
Left channel Sample #n															
00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
Right channel Sample #n															

Table 6: RTP Payload Format – 16 Bit Stereo

Note: Except for the sampling frequency, this payload type is identical to payload type 10 (L16, stereo, 44.1 KHz), as defined in RFC 3551 [11].

5.3 Establishing the RTP Connection

RTP streaming in MirrorLink is based on UDP, which is connectionless. Some level of initial connection is REQUIRED for the MirrorLink RTP Server to know the MirrorLink RTP Client's IP address and port number.

5.3.1 RTP Server within MirrorLink Server

When the MirrorLink Server takes on the role of the MirrorLink RTP server, the MirrorLink Client MUST send an UDP packet, containing 1 byte[8] to the port number and IP address assigned for the MirrorLink server's

[8] The content of the byte can be any arbitrary value.

audio server. On reception of that UDP packet, the audio server MUST determine the IP address and port number of the audio client.

After receiving that packet, the audio server within the MirrorLink Server's RTP Server will start RTP streaming to the RTP Client using the port number, for the received 1 byte packet, when audio stream is available.

The message flow, as shown in Figure 4, consists of the following steps:

1. RTP server: Wait for a UDP packet with server's start.
2. RTP client: Send 1 byte UDP packet to the RTP server; the RTP server address is obtained through the TmApplicationServer UPnP service.
3. RTP client: Get ready for receiving RTP packets.
4. RTP server: Determine client's IP address and port number.
5. RTP server: Begin RTP streaming; if no audio data is available, the RTP server SHOULD send a single RTP packet, without data payload
6. RTP client receives RTP packets.
7. RTP client: If the RTP client does not receive any RTP packets it MUST continue sending 1 byte UDP packets to the RTP server (as defined in step 2) at regular time intervals.

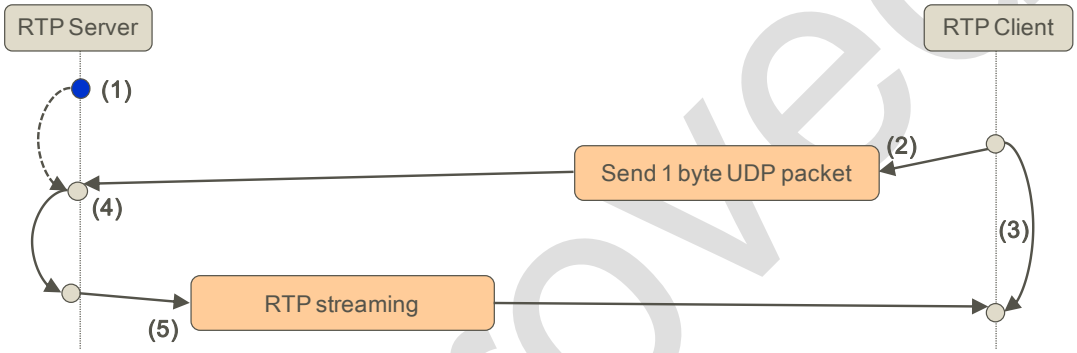


Figure 4: Sequence to Establish the RTP Connection

5.3.2 RTP Server within MirrorLink Client

When the MirrorLink Client takes the role of the MirrorLink RTP Server, the MirrorLink Client MUST use the destination IP address and port number from the UPnP LaunchApplication action's response. Therefore there is no need to wait for the 1 byte UDP packet.

The MirrorLink Client's RTP server will start RTP streaming to the RTP client, when audio stream becomes available.

The following sequences show how RTP streaming for a MirrorLink Client based RTP server works:

1. UPnP Control Point: Invoke UPnP LaunchApplication action for the respective RTP Client on the MirrorLink Server.
2. UPnP Server: Return IP address and port number for the RTP client as a reply for the LaunchApplicationRequest action.
3. RTP Client: Get ready for getting RTP streaming.
4. RTP Server: Starts RTP streaming; if no audio data available, RTP Server SHOULD send a single RTP packet, without data payload.

5.4 Client and Server Implementation

For each audio stream, the audio client SHOULD do local buffering of the incoming RTP packets and SHOULD start playing the audio, if the buffer is filled with a reasonable number of packets. This will allow compensating for potential jittering.

The audio client MUST compensate potential frequency differences between the server and client audio sampling rate.

The audio server is responsible for maintaining long term accuracy of the audio clock. For example, if a 48kHz audio stream is sent over 10 seconds the RTP server SHOULD make sure that all data are delivered around 10 seconds' duration with minimal error. Note that each packet can arrive at different intervals depending on the network load, and a reasonable amount of buffering SHOULD solve this problem. The audio client is expected to use its own audio clock to play the received audio stream, and it is the audio client's responsibility to compensate potential frequency differences between the server and client audio clock. When an audio server supplies audio stream in much faster frequency than audio client can handle, the audio client can either drop some packets or adjust its playback frequency. It is up to client to decide which method to choose.

It is expected that the audio is played immediately, without any delays, besides the buffering delay. No specific synchronization between the audio stream and the VNC based display updates is provided.

The RTP client SHOULD provide the following RTP streaming parameter to the RTP server, to allow uninterrupted audio playback, even if the audio transfer is subject to network jitter;

1. Initial Playback Latency (IPL) [samples]

   If the audio buffer has been empty, the RTP client MUST start the audio playback only when the audio buffer is filled with IPL audio samples (the IPL value is based on RTP payload type 99).

   For other payload types, the new IPL is calculated according the following formula

$$IPL_{payloadType} = floor\left(IPL_{payloadType=99} \cdot \frac{f_{payloadType}}{48kHz} \right)$$

   The default IPL value is 4800.

   The RTP Client MAY use a separate low-latency buffer for conversational audio, rather than the application audio buffer as defined with audioIPL. Conversational audio over RTP MUST use the RTP header extension with appCategory = 0xF0000020 (Conversational Audio). Voice Command audio over RTP MUST use the RTP header extension with appCategory = 0xF0000010 (Voice Command Engine).

   Note: The RTP Client SHOULD pause the audio when RTP packet with Marker bit set to 1 received. The RTP Client SHOULD discard the audio buffer content, if audio playback cannot be started or resumed within 1s after receiving the first RTP packet.

   The IPL value MUST provide sufficient cover to include any potential delay, originating from the MirrorLink Client's need to ramp-up the MirrorLink audio chain.

2. Maximum Payload Length (MPL) [samples]

   The RTP client MUST be able to store MPL audio samples into the audio buffer (the MPL value is based on RTP payload type 99).

   For other payload types, the new MPL is calculated according the following formula

$$MPL_{payloadType} = floor\left(MPL_{payloadType=99} \cdot \frac{f_{payloadType}}{48kHz} \right)$$

   The default MPL value is 9600.

The RTP server SHOULD send RTP packets, on average, at regular time intervals, equal to the time it takes to play them at the RTP client. The RTP server MUST NOT cause a buffer underrun or overflow at the RTP client. The RTP server MAY initially send a bigger packet to allow for compensation of future network jitter. The RTP server is responsible for compensating for expected network jitter, by ensuring that enough audio samples are transmitted to the client.

The RTP Server MUST set the M-bit to 1 if no further audio is currently available for streaming, i.e. the RTP Server expects the RTP Client's pipeline getting fully emptied, prior potentially resuming the RTP streaming

at a later time. This may happen due to e.g. pausing or terminating the audio source[9]. Note: The M-Bit cannot necessarily be used to identify the end of an individual audio source.

RTP Client SHOULD dispose audio content, if the RTP Server suspends audio streaming (M bit set to 1) or changes the RTP payload type, and if the RTP Client has not started the audio playback.

The MirrorLink client SHOULD provide the RTP streaming parameters within the TmClientProfile service as defined in [14]. The MirrorLink server SHOULD provide the RTP streaming parameters within the A_ARG_TYPE_AppList response as defined in [13].

The MirrorLink Server MUST set the system's audio volume to a reasonable level and MUST prevent the user from changing the audio volume during a MirrorLink session[10]. Note: The specification does not prevent applications, to intentionally provide an audio stream with a low audio volume.

The MirrorLink Server MUST only send RTP packets, if they carry real audio content.

[9] Example: To indicate the end of a navigation voice guidance, if no other audio sources are playing.

[10] Example: Changing the audio volume via the MirrorLink Server's UI or external volume buttons

6 VOICE COMMAND HANDLING

6.1 Voice Command over RTP with Established VNC Session

Voice Command (VC) over RTP sessions, consisting of one or more individual voice commands, are facilitated using the VNC Device Status message's Voice Command and Microphone flags.

There are the following basic requirements for MirrorLink Clients and Servers about how to trigger (i.e. start and terminate) a VC session over RTP:

- A MirrorLink Server MUST support triggering a VC session from the MirrorLink Client and Server.
- A MirrorLink Client MUST support triggering a VC session from the MirrorLink Server.
- A MirrorLink Client SHOULD support triggering a VC session from the MirrorLink Client.

The Device Status Voice Command flag indicates the start and end of a VC session, whereas the Device Status Microphone flag indicates the start and end of individual Voice commands. In case of a Voice Command session, consisting of a single Voice Command, both flags can be set at the same time. The basic flow is shown in Figure 5.

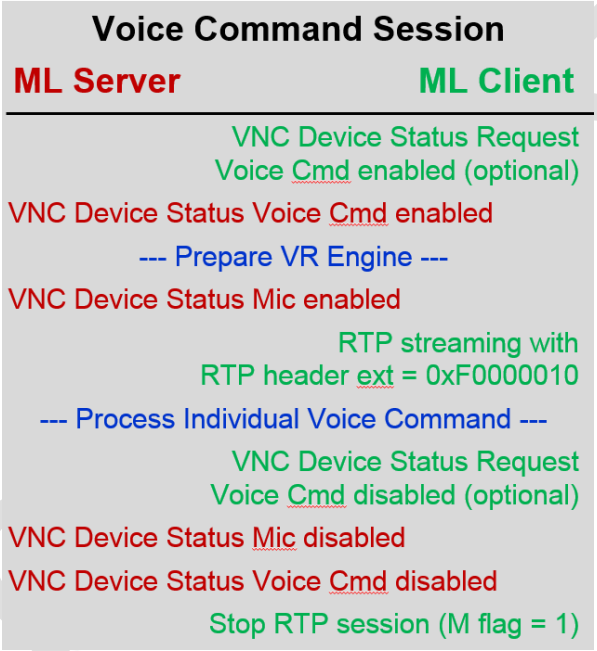


Figure 5: Voice Command Session over RTP

The MirrorLink Client MAY request a VC Session via VNC Device Status Request with Voice Command enabled, e.g. triggered by PTT button. The MirrorLink Server decides whether VC is supported and available and MUST responds with VNC Device Status Voice Command enabled or disabled. In case the VC session is started from the MirrorLink Server, the MirrorLink Server MUST send the VNC Device Status Voice Command enabled message, without a prior Device Status Request message.

While the VC session is active, the MirrorLink Client MAY inform the user via e.g. pop-up, if MirrorLink screen is not in foreground.

The MirrorLink Server MUST send a VNC Device Status Microphone enabled message to request microphone input, when the MirrorLink Server is ready to process the VC. The MirrorLink Client MUST open the microphone, when the VNC Device Status Microphone enabled message has been received AND the Microphone is available at the MirrorLink Client. Then, the MirrorLink Client MUST start the RTP streaming, as soon as voice command input is available. While the VC session is active, the MirrorLink Client MUST send the Voice Command audio with RTP header extension equal 0xF0000010 (otherwise 0xF0000020).

The MirrorLink Client MUST respond to a VNC Device Status Microphone enabled message, with a VNC Device Status Request Microphone disabled message if the Microphone is currently occupied.

The MirrorLink Server MUST stop microphone input by sending VNC Device Status Voice Mic disabled message. MirrorLink Server or Client MUST stop VC session by sending VNC Device Status Request message with the Voice Command flag disabled.

The VC engine on the MirrorLink Server MAY request to continue the VC dialog, sending a new VNC Device Status message with the Microphone flag enabled. The Dialog phase is repeated then, as shown in Figure 6.

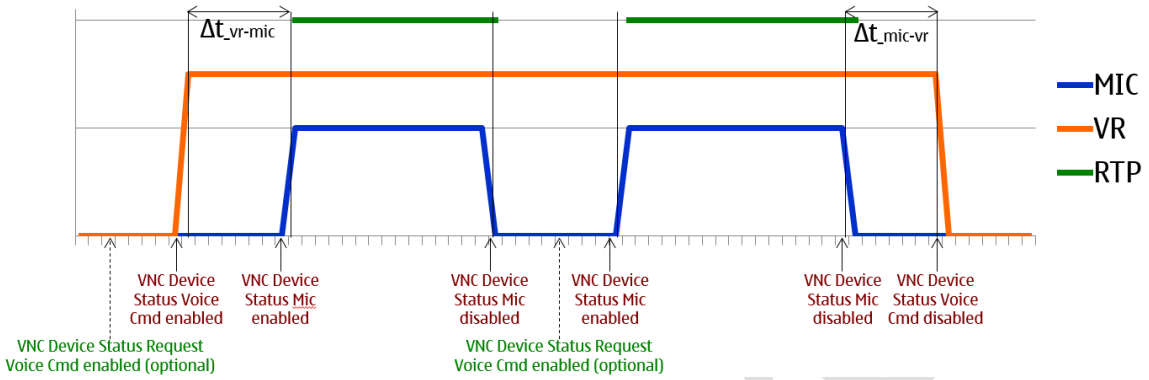


Figure 6: Voice Command Dialog over RTP

The MirrorLink Server SHOULD send VNC Device Status with the Voice Command flag enabled together with or before Microphone flag enabled ($\Delta t\_vr\text{-}mic \geq 0$)

The MirrorLink Server SHOULD send VNC Device Status with the Voice Command flag disabled together with or after Microphone disabled ($\Delta t\_mic\text{-}vr \geq 0$)

A VC session MAY take place during a Phone Call Session, as shown below in Figure 7

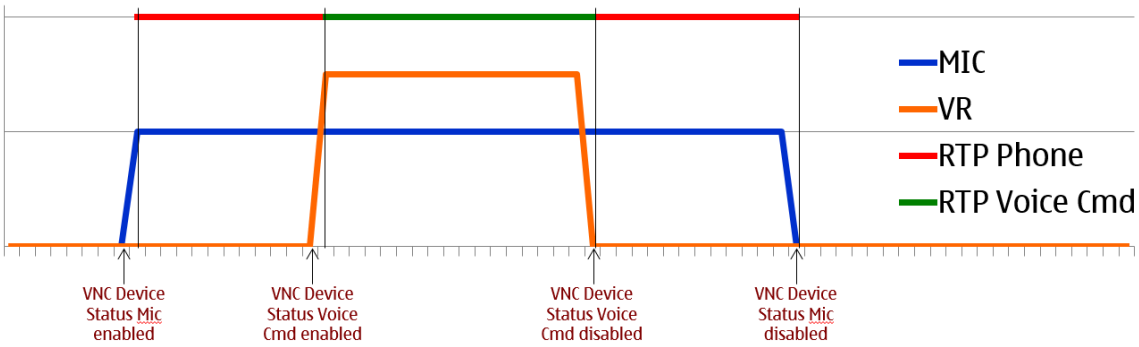


Figure 7: Voice Command Session over RTP with ongoing Phone Call

7 INTEROPERABILITY WITH BLUETOOTH

Interoperability (IOP) of the MirrorLink RTP streaming audio framework with Bluetooth audio profiles MUST be given. Bluetooth can be used to replace missing or existing RTP audio streaming functionality as defined in this MirrorLink specification.

The MirrorLink server MUST at least distinguish between speech (phone call) and application audio (media player, navigation directions, etc.).

The MirrorLink server SHOULD provide BT HFP, if it supports phone call or voice control functionality.[11]

7.1 Bluetooth Profiles relevant for MirrorLink

Bluetooth Headset Profile (BT HSP)

Bluetooth Headset profile (BT HSP) MUST NOT be used in MirrorLink.

Bluetooth Hands-Free Profile (BT HFP)

If Bluetooth Hands-free profile (BT HFP) is used for voice call together with RTP streaming for application audio, it is the responsibility of the MirrorLink server to take control of audio link (SCO) when phone call is active: The MirrorLink server will take care of both opening and closing the SCO audio link. The MirrorLink client will accept the setup of the SCO audio link. [12] It is the MirrorLink client's responsibility to make sure that the SCO audio link is opened as soon as possible when requested from the MirrorLink server. Except the explicit MirrorLink server's responsibility for the audio link control, Bluetooth Hands-Free Profile 1.5 specification will be followed.

Bluetooth Advanced Audio Distribution Profile (BT A2DP)

Bluetooth A2DP MAY optionally be used to stream application audio from the device to the MirrorLink client. If RTP streaming is available, it is not RECOMMENDED to use BT A2DP for application audio.

7.2 Interoperability States –MirrorLink Server Perspective

Within the MirrorLink context, the following Interoperability (IOP) states with Bluetooth (BT) can be distinguished, as shown in Figure 8 from the CE MirrorLink server, i.e. the CE device perspective.

- MirrorLink Server has established neither any MirrorLink and nor any legacy[13] Bluetooth connections (**State: None**)
- MirrorLink Server has established a legacy Bluetooth connection to the vehicle head-unit, but no MirrorLink connection (**State: BT to HU**)
- MirrorLink Server has established a legacy Bluetooth connection to a head-set (**State: BT to HS**)
- MirrorLink Server has established a MirrorLink connection to the MirrorLink Client (**State: ML to HU**)

[11] The MirrorLink server cannot assume the client supports bi-directional RTP streaming.

[12] There is no difference for the head-unit, whether the audio link is controlled from a phone application (without MirrorLink being active) or from the MirrorLink server.

[13] Legacy Bluetooth connections are handled outside MirrorLink.

- MirrorLink Server has established a MirrorLink connection and a legacy Bluetooth connection to the MirrorLink Client (**State: ML to HU + BT to HU**)
- Mobile device has established a Bluetooth connection to a head-set in addition to a MirrorLink connection (**State: ML to HU+ BT to HS**)

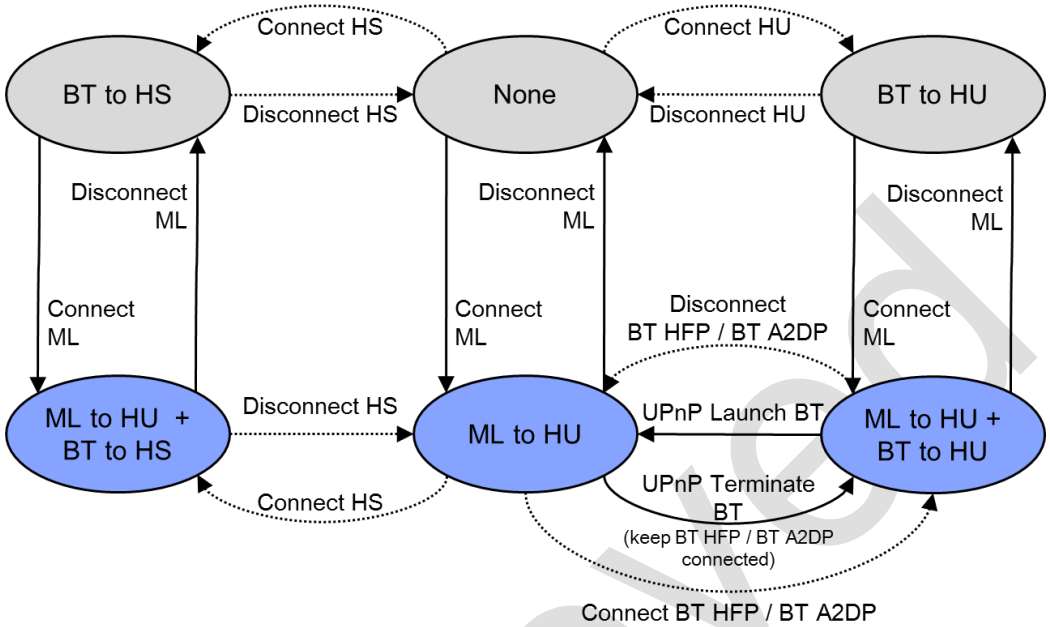


Figure 8: Bluetooth / MirrorLink IOP States – MirrorLink Server Perspective

The transitions between the different IOP states MUST be followed from actions with regard to the head-set (HS) or head-unit (HU) Bluetooth profiles or the RTP streaming. These actions are given in Table 7. Transitions marked in dotted lines, are outside the scope of this specification and given for completeness only.

Current State	Action	Next State	Immediate Activity for MirrorLink Server Connectivity			
			HS BT HFP	HU BT HFP	HU BT A2DP	HU RTP stream
None	Connect ML	ML to HU	-	Connect (*)	Connect (*)	Connect (*)
	Connect HS	BT to HS	Connect	-	-	-
	Connect HU	BT to HU	-	Connect (**)	Connect (**)	-
BT to HS	Connect ML	ML to HU + BT to HS	Keep	Reject	Reject	Connect (*)
	Disconnect HS	None	Disconnect	-	-	-
BT to HU	Connect ML	ML to HU + BT to HU	-	Keep	Keep	Connect (*)
	Disconnect BT	None	-	Disconnect	Disconnect	-
ML to HU	Disconnect ML	None → BT to HU	-	Keep	Keep	Disconnect
	Connect BT	ML to HU + BT to HU	-	Connect (**)	Connect (**)	Keep
	UPnP Terminate BT	ML to HU + BT to HU	-	Keep	Keep	Keep
	Connect HS	ML to HU + BT to HS	Connect	Disconnect	Disconnect	Keep
	Disconnect ML	BT to HU	-	Keep	Keep	Disconnect

Current State	Action	Next State	Immediate Activity for MirrorLink Server Connectivity			
			HS BT HFP	HU BT HFP	HU BT A2DP	HU RTP stream
ML to HU + BT to HU	UPnP Launch BT	ML to HU	-	Keep	Keep	Keep
	Disconnect BT	ML to HU	-	Disconnect	Disconnect	Keep
ML to HU + BT to HS	Disconnect HS	ML to HU	Disconnect	Connect (*)	Connect (*)	Keep
	Disconnect ML	BT to HS	Keep	-	-	Disconnect

Table 7: IOP Transition (from MirrorLink Server perspective)

(*) If selected during MirrorLink UPnP negotiation

(**) If selected during legacy BT pairing

The connection to / disconnection from a wired audio accessory SHOULD be treated the same way as the connection / disconnection from a Bluetooth head-set (BT HS).

7.3 Interoperability States –MirrorLink Client Perspective

Within the MirrorLink context, the following Interoperability (IOP) states with Bluetooth (BT) can be distinguished, as shown in Figure 9 from the MirrorLink client, i.e. Head-Unit perspective.

- MirrorLink Client has established neither any MirrorLink and nor any legacy Bluetooth connections (**State: None**)
- MirrorLink Client has established a Bluetooth connection to a mobile device, but no MirrorLink connection (**State: BT to CE1**)
- MirrorLink Client has established a Bluetooth connection to a mobile device, different from CE1 (**State: BT to CE2**)
- MirrorLink Client has established a MirrorLink connection with CE1 (**State: ML to CE1**)
- MirrorLink Client has established a MirrorLink connection and a legacy Bluetooth connection to the CE1 (**State: ML to CE1 + BT to CE1**)
- MirrorLink Client has established a Bluetooth connection to CE2 in addition to a MirrorLink connection with CE1 (**State: ML to CE1 + BT to CE2**)

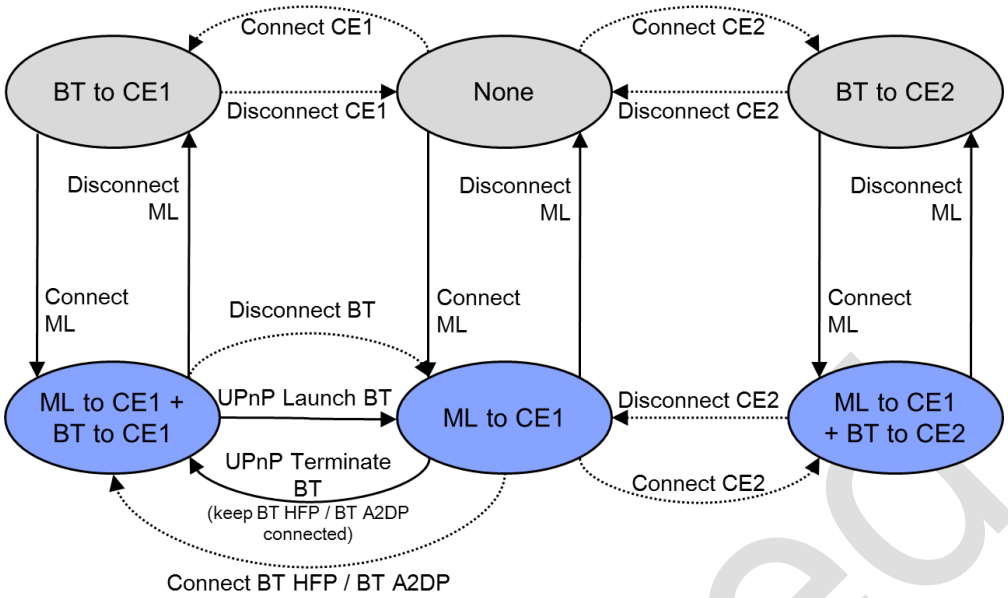


Figure 9: Bluetooth / MirrorLink IOP States – MirrorLink Client Perspective

The transitions between the different IOP states MUST be followed from actions with regard to the CE1 and CE2 Bluetooth profiles or the RTP streaming. These actions are given in Table 8. Transitions marked in dotted lines, are outside the scope of this specification and given for completeness only.

Current State	Action	Next State	Immediate Activity for MirrorLink Client Connectivity			
			CE2 BT HFP	CE1 BT HFP	CE1 BT A2DP	CE1 RTP stream
None	Connect ML	ML to CE1	-	Connect (*)	Connect (*)	Connect (*)
	Connect CE2	BT to CE2	Connect	-	-	-
	Connect CE1	BT to CE1	-	Connect (**)	Connect (**)	-
BT to CE2	Connect ML	ML to CE1 + BT to CE2	Keep	Reject	Reject	Connect (*)
	Disconnect CE2	None	Disconnect	-	-	-
BT to CE1	Connect ML	ML to CE1 + BT to CE1	-	Keep	Keep	Connect (*)
	Disconnect CE1	None	-	Disconnect	Disconnect	-
ML to CE1	Disconnect ML	None → BT to CE1	-	Keep	Keep	Disconnect
	Connect BT	ML to CE1 + BT to CE1	-	Connect (**)	Connect (**)	Keep
	UPnP Terminate BT	ML to CE1 + BT to CE1	-	Keep	Keep	Keep
	Connect CE2	ML to CE1 + BT to CE2	Connect	Disconnect	Disconnect	Keep
ML to CE1 + BT to CE1	Disconnect ML	BT to CE1	-	Keep	Keep	Disconnect
	UPnP Launch BT	ML to CE1	-	Keep	Keep	Keep
	Disconnect BT	ML to CE1	-	Disconnect	Disconnect	Keep

Current State	Action	Next State	Immediate Activity for MirrorLink Client Connectivity			
			CE2 BT HFP	CE1 BT HFP	CE1 BT A2DP	CE1 RTP stream
ML to CE1 + BT to CE2	Disconnect CE2	ML to CE1	Disconnect	Connect (*)	Connect (*)	Keep
	Disconnect ML	BT to CE2	Keep	-	-	Disconnect

Table 8: IOP Transition (from MirrorLink Client perspective)

(*) If selected during MirrorLink UPnP negotiation

(**) If selected during legacy BT pairing

The selection of the audio link is specified within section 3.

8 AUDIO SIGNAL PROCESSING CONFIGURATION

8.1 Conversational and Telephony-based Audio

In order to optimize perceived audio quality during conversational audio sessions in MirrorLink, clients and servers SHOULD conform to audio performance specifications and behaviors as defined in ITU-T-P.1100. To achieve this, the MirrorLink server SHOULD disable all audio signal processing functions supported when a MirrorLink connection exists. The MirrorLink client SHOULD perform all audio signal processing functions in the system for conversational audio.

8.2 Bluetooth HFP Noise Reduction/Echo Cancellation

In HFP specification, all audio signal processing in the MirrorLink server SHOULD be disabled in correlation with the HFP requirements specifying AG Noise Reduction/Echo Cancellation.[14] The server SHOULD re-enable audio signal processing functions upon audio transfer to the MirrorLink server device, or disconnection of HFP.

The MirrorLink client SHOULD implement HFP feature to disable audio signal processing, and implement appropriate local audio processing for conversational and telephony audio.

8.3 RTP-based Conversational Audio

If a MirrorLink client establishes an RTP connection for call or conversational audio usage, the MirrorLink server SHOULD disable all local audio signal processing. The server SHOULD re-enable audio signal processing functions upon disconnection of RTP for conversational audio (i.e. call audio transfer to the MirrorLink server device), or internal detection of idle call state.

If the MirrorLink client establishes an RTP connection for conversational audio usage, it SHOULD implement appropriate local audio processing for conversational and telephony audio.

The RTP Server and Client SHOULD follow ITU-T G.114 recommendation [15] for conversational audio.

[14] It is assumed that a MirrorLink server always perform the role of Hands-Free Profile Audio Gateway, as this is the primary way to provide hands-free calling with an in-vehicle infotainment system.

9 REFERENCES

[1] IETF, RFC 3550, "RTP: A Transport Protocol for Real-Time Applications", July 2003, http://tools.ietf.org/html/rfc3550

[2] BT Specification, "Hands-free Profile 1.5", Car Working Group, Revision V10r00, November 25, 2005.

[3] BT Specification, "Headset Profile", Car Working Group, Revision V12r00, December 18, 2008.

[4] UPnP Specification, "UPnP™ Device Architecture", Version 1.1, Revision October 15, 2008

[5] Bluetooth Assigned Numbers, http://www.bluetooth.org/assigned-numbers.htm

[6] Bluetooth Specification "Audio/Video Distribution Transport Protocol (AVDTP)", Revision 1.00

[7] IETF, RFC 2119, "Keys words for use in RFCs to Indicate Requirement Levels", March 1997. http://www.ietf.org/rfc/rfc2119.txt

[8] IETF, RFC 2190, "RTP Payload Format for H.263 Video Streams", September 1997, http://tools.ietf.org/html/rfc2190

[9] IETF, RFC 4648, "The Base 16, Base 32 and Base 64 Data Encodings", October 2006. http://tools.ietf.org/html/rfc4648

[10] IETF, RFC 5280, "Internet X.509 Public Key Infrastructure Certificate", May 2008. http://tools.ietf.org/html/rfc5280

[11] IETF, RFC 3551, "RTP Profile for Audio and Video Conferences with Minimal Control", July 2003, http://tools.ietf.org/html/rfc3551

[12] Car Connectivity Consortium, "MirrorLink – UPnP Server Device", Version 1.1; CCC-TS-030

[13] Car Connectivity Consortium, "MirrorLink – UPnP Application Server Service", Version 1.1; CCC-TS-024

[14] Car Connectivity Consortium, "MirrorLink – UPnP Client Profile Service", Version 1.1; CCC-TS-026

[15] ITU-T Rec. G.114 (05/2003), "One-way transmission time", May 6, 2003; available from http://www.itu.int

[16] Car Connectivity Consortium, "MirrorLink – Common API", Version 1.1.0, CCC-TS-038