Car Connectivity Consortium

MirrorLink®

MirrorLink over Wi-Fi Display

Version 1.2.3
(CCC-TS-049)

VERSION HISTORY

Version	Date	Comment
1.2.0	25 September 2013	Approved Version
1.2.1	23 April 2014	Approved Errata Version
1.2.2	10 November 2014	Approved Errata Version
1.2.3	14 May 2015	Approved Errata Version

LIST OF CONTRIBUTORS

Bai, Fan	General Motors
Benesch, Matthias	Mercedes-Benz Research & Development North America
Brakensiek, Jörg (Editor)	Microsoft Corporation
Jativa-Villoldo, Juan V.	Nokia
Sungjin, Park	Samsung Electronics
Kafle, Padam	Qualcomm
Shaukat, Fawad	Qualcomm
Subramaniam, Vijay	Qualcomm
Raveendran, Viji	Qualcomm

LEGAL NOTICE

The copyright in this Specification is owned by the Car Connectivity Consortium LLC (“CCC LLC”). Use of this Specification and any related intellectual property (collectively, the “Specification”), is governed by these license terms and the CCC LLC Limited Liability Company Agreement (the “Agreement”).

Use of the Specification by anyone who is not a member of CCC LLC (each such person or party, a “Member”) is prohibited. The legal rights and obligations of each Member are governed by the Agreement and their applicable Membership Agreement, including without limitation those contained in Article 10 of the LLC Agreement.

CCC LLC hereby grants each Member a right to use and to make verbatim copies of the Specification for the purposes of implementing the technologies specified in the Specification to their products (“Implementing Products”) under the terms of the Agreement (the “Purpose”). Members are not permitted to make available or distribute this Specification or any copies thereof to non-Members other than to their Affiliates (as defined in the Agreement) and subcontractors but only to the extent that such Affiliates and subcontractors have a need to know for carrying out the Purpose and provided that such Affiliates and subcontractors accept confidentiality obligations similar to those contained in the Agreement. Each Member shall be responsible for the observance and proper performance by such of its Affiliates and subcontractors of the terms and conditions of this Legal Notice and the Agreement. No other license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Any use of the Specification not in compliance with the terms of this Legal Notice, the Agreement and Membership Agreement is prohibited and any such prohibited use may result in termination of the applicable Membership Agreement and other liability permitted by the applicable Agreement or by applicable law to CCC LLC or any of its members for patent, copyright and/or trademark infringement.

THE SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE,NONINFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS, AND COMPLIANCE WITH APPLICABLE LAWS.

Each Member hereby acknowledges that its Implementing Products may be subject to various regulatory controls under the laws and regulations of various jurisdictions worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation and distribution of Implementing Products. Examples of such laws and regulatory controls include, but are not limited to, road safety regulations, telecommunications regulations, technology transfer controls and health and safety regulations. Each Member is solely responsible for the compliance by their Implementing Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Implementing Products related to such regulations within the applicable jurisdictions.

Each Member acknowledges that nothing in the Specification provides any information or assistance in connection with securing such compliance, authorizations or licenses**.**

NOTHING IN THE SPECIFICATION CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS. ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY INTELLECTUAL PROPERTYRIGHTS OR FOR NONCOMPLIANCE WITH LAWS, RELATING TO USE OF THE SPECIFICATION IS EXPRESSLY DISCLAIMED. BY USE OF THE SPECIFICATION, EACH MEMBER EXPRESSLY WAIVES ANY CLAIM AGAINST CCC LLC AND ITS MEMBERS RELATED TO USE OF THE SPECIFICATION.

CCC LLC reserve the right to adopt any changes or alterations to the Specification as it deems necessary or appropriate.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TERMS AND ABBREVIATIONS

ABC	Audio Back Channel
HDCP	High-bandwidth Digital Content Protection
IE	Information Element
IEEE	Institute of Electrical and Electronics Engineers
Miracast	Commercial denomination of WFD
ML	MirrorLink
OUI	Organizationally Unique Identifier
PES	Packetized Elementary Stream
Sink Device	A device that receives multimedia content from a WFD source over a Wi-Fi link and renders it.
Source Device	A device that supports streaming multimedia content to a WFD sink(s) over a Wi-Fi link.
UIBC	User Input Back Channel
UPnP	Universal Plug and Play
USB	Universal Serial Bus
VNC	Virtual Network Computing
WFD	Wi-Fi Display which is the technology and specification being officially called “Wi-Fi Alliance Wi-Fi Display specification”

HDCP is a registered trademark of Digital Content Protection LLC.

Miracast is a registered trademark of the Wi-Fi Alliance.

MirrorLink is a registered trademark of Car Connectivity Consortium LLC.

UPnP is a registered trademark of UPnP Implementers Corporation.

VNC is registered trademarks of RealVNC Ltd.

Other names or abbreviations used in this document may be trademarks of their respective owners.

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 car head-unit to interact with the mobile device in mind, but it will similarly apply for other devices, which do provide a colored display, audio input/output and user input mechanisms.

This specification describes the integration of Wi-Fi Display to MirrorLink.

The specification lists a series of requirements, either explicitly or within the text, which are mandatory elements for a 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 are following the notation as described in RFC 2119 [6].

1. MUST: This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.
2. MUST NOT: This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.
3. SHOULD: This word, or the adjective "RECOMMENDED", mean 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" mean 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

Wi-Fi Display, also known as Miracast, is a peer-to-peer wireless screen replication standard created by the Wi-Fi Alliance. Its main purpose is to let the source device project its screen to the sink device screen, and to provide the sink device with the method to control the source device.

Figure 1 shows the typical Client/Server topology for the MirrorLink over Wi-Fi Display.

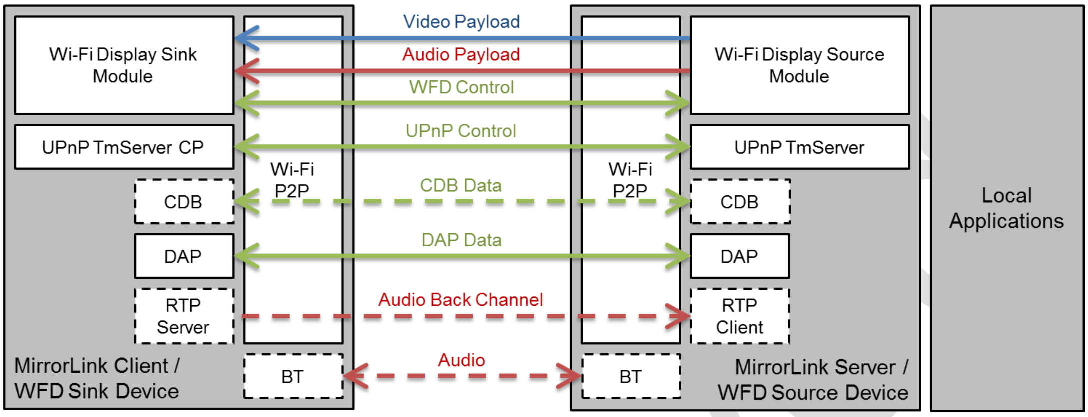

Figure 1: High Level Topology

This document specifies the integration of Wi-Fi Display into MirrorLink, providing an alternative video link to VNC. The specification of the other MirrorLink components, like UPnP, CDB, DAP etc. is done in their respective documents.

The MirrorLink Client, providing WFD functionality, MUST implement the WFD Sink functionality.

The MirrorLink Server, providing WFD functionality, MUST implement the WFD Source functionality.

Note: The term "Sink" used in this specification refers to a WFD Primary Sink device as defined in [5].

Figure 2 displays the layered architecture diagram for the integration of WFD into MirrorLink. WFD stack is added to MirrorLink stack. The diagram applies to both Client and Server devices, which must apply it according to their roles.

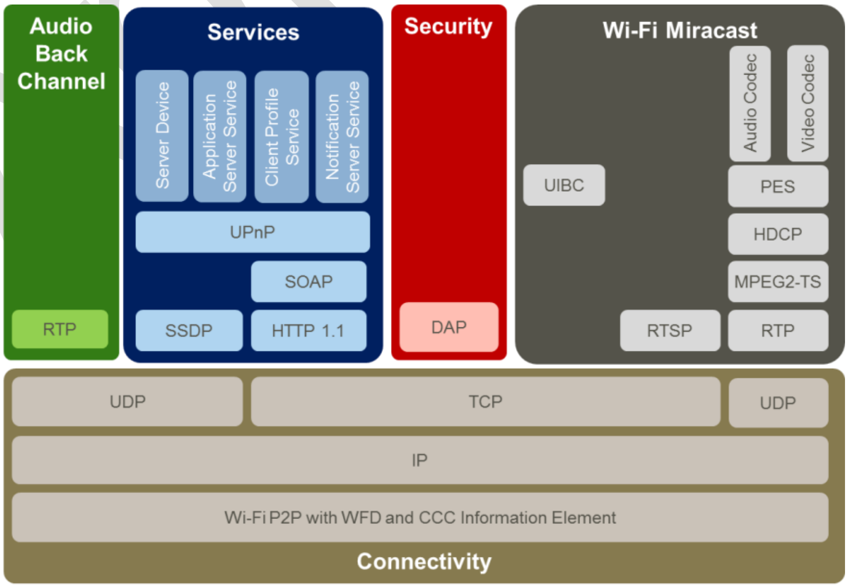

Figure 2: MirrorLink over Wi-Fi Display Architecture

Through Wi-Fi Display, MirrorLink Server and Client discover each other with the Wi-Fi Device Discovery procedure, which exchange the Information Element. WFD UIBC is integrated into MirrorLink stack for User Input.

MirrorLink Client and Server MUST support all Wi-Fi Display mandatory functions and services, as described in [5], Table 3-1. This includes the following functions and services:

- WFD Device Discovery with IE for CCC
- WFD Connection Setup
- WFD Capability negotiation
- WFD Session establishment
- Encoding and packetization of the captured Display
- Transport of multiplexed audio and video payload
- De-multiplex, de-packetization and decode of received audio and video payload
- Rendering of decoded video on local display panel
- Power Save mechanisms
- Session termination
- Encode and packetization of captured audio
- Multiplex video and audio payload
- Rendering of decoded audio on local speakers
- AV Stream Control using RTSP

The MirrorLink Client and Server MUST support the following optional Wi-Fi Display functions:

- User Input Back Channel (UIBC)

Use of BT HFP in accordance with the MirrorLink Audio Specification [2] SHALL be possible for the MirrorLink over WFD implementation as well.

3 MIRRORLINK OVER WFD PROCEDURE

MirrorLink over Wi-Fi Display (WFD) connection between MirrorLink Server acting as WFD source device and MirrorLink Client acting as WFD sink device MUST take place in the 4 following phases, as depicted in Figure 3.

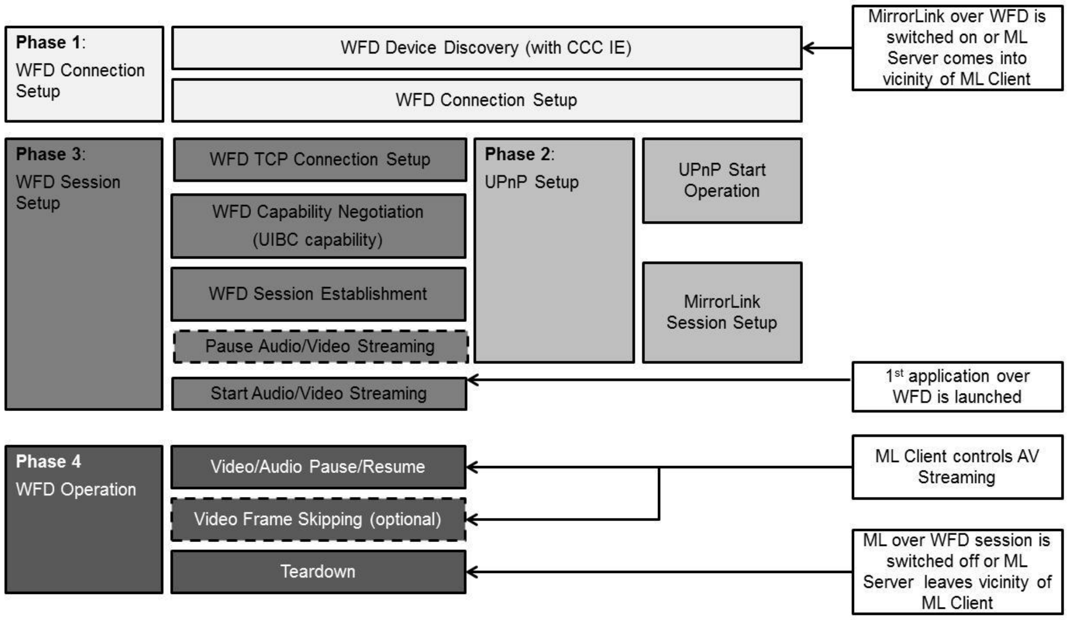
Phase 1:
WFD Connection Setup
WFD Device Discovery (with CCC IE)
MirrorLink over WFD is switched on or ML Server comes into vicinity of ML Client
WFD Connection Setup
Phase 3:
WFD Session Setup
WFD TCP Connection Setup
WFD Capability Negotiation (UIBC capability)
WFD Session Establishment
Pause Audio/Video Streaming
Start Audio/Video Streaming
Phase 2:
UPnP Setup
UPnP Start Operation
MirrorLink Session Setup
1st application over WFD is launched
Phase 4
WFD Operation
Video/Audio Pause/Resume
Video Frame Skipping (optional)
Teardown
ML Client controls AV Streaming
ML over WFD session is switched off or ML Server leaves vicinity of ML Client

Figure 3: MirrorLink over Wi-Fi Display Diagram

3.1 Phase 1: WFD Connection Setup

Phase 1 MUST start when MirrorLink over Wi-Fi Display is switched on. In addition, if persistent WFD Group for MirrorLink exists, it is recommended that WFD Connection setup proceeds automatically without user interaction such as re-selection of WFD & CCC capable device.

The following requirements apply to the phase 1:

WFD Device Discovery:

To establish a MirrorLink over Wi-Fi Display connection, Wi-FiP2P device discovery with WFD IE (Information Element) MUST be used. Wi-Fi Display devices MUST advertise the WFD IEs defined in Wi-Fi Display specification.

In addition to the WFD IEs, the MirrorLink devices MUST include the CCC Information Element that MUST contain the MirrorLink UPnP Device Information sub-element and MAY contain the Internet Accessibility sub-element, as specified in [3], into all beacon, probe request and probe response frames. The MirrorLink devices MUST detect other MirrorLink devices through CCC IE.

Note

- WFD connection using Wi-Fi P2P MUST be supported. A WFD connection using TDLS is OPTIONAL.

WFD Connection Setup

The MirrorLink devices MUST follow the process of Wi-Fi P2P/WFD as specified in [5].

The MirrorLink devices MUST connect to a device, which includes a WFD IE and a CCC IE. To establish a P2P connection for a WFD connection setup, the MirrorLink devices MUST also include the CCC Information Element that MUST contain the MirrorLink UPnP Device Information sub-element and MAY contain the Internet Accessibility sub-element, as specified in [3], when transmitting the P2P Invitation Request, P2P Invitation Response, GO Negotiation Request, GO Negotiation Response, GO Confirmation, Association/Reassociation Request and Association/Reassociation Response frames.

The Persistence WFD Group allows automatic WFD connection through caching the information for the Group. To establish a Persistence WFD Group, the MirrorLink devices SHOULD follow the process of WFD as specified in [5].

3.2 Phase 2: UPnP Setup

Phase 1 MUST be completed, before phase 2 can start.

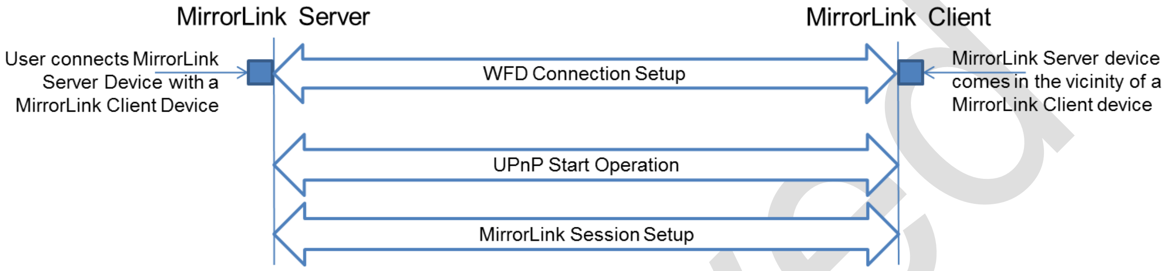
MirrorLink Server
MirrorLink Client
User connects MirrorLink Server Device with a MirrorLink Client Device
MirrorLink Server device comes in the vicinity of a MirrorLink Client device
WFD Connection Setup
UPnP Start Operation
MirrorLink Session Setup

Figure 4: UPnP Setup Sequence Diagram

UPnP Operation Start

Based on the information within the CCC IE from the WFD device discovery and WFD connection setup stages, the MirrorLink device activates the respective UPnP components (i.e. TmServer Control Point or TmServerServer) in the device.

The MirrorLink Client MUST immediately activate its UPnP TmServerDevice:1 Control Point if it connects to a MirrorLink device with a UPnP TmServerDevice:1 Server.

The MirrorLink Server MUST immediately activate its UPnP TmServerDevice:1 Server, if it connects to a MirrorLink device with a UPnP TmServerDevice:1 Control Point.

The MirrorLink Client MUST follow the UPnP Operation Start sequence defined in the MirrorLink Core Architecture specification [8].

UPnP Device Description

The MirrorLink Client devices MUST either use the CCC Information Element, defined in [3], or MUST send an `SSDP:discover` message to determine the location of the MirrorLink Server's UPnP Server Device XML. The MirrorLink Client MUST NOT wait for an `SSDP:alive` messages for initial UPnP Setup.

The structure of the UPnP Server Device XML is specified in [4]. The UPnP Server Device XML description MUST be accessible at the following URL via HTTP-GET.

```
http://<IPAddress>:<Port Number>/<path>
```

The MirrorLink Client MUST form the UPnP Device Description URL with the following elements, when using the CCC Information Element:

- `IP Address`: IP address of the MirrorLink Server, as retrieved with in the DHCP negotiation with the WFD connection setup process. The DHCPOFFER message includes both, the DHCP Server and Client IP address.
- `Port Number`: Port number of the MirrorLink UPnP Server, as provided in the MirrorLink UPnP Device Information sub-elements of the CCC Information Element as specified in [3].

- o `Path`: Static path "`TmServerDevice/TmServerDevice:1.xml`".

Example

```
http://192.168.3.15:2869/TmServerDevice/TmServerDevice:1.xml
```

The MirrorLink Server MUST provide the same URL to its Device XML via the CCC Information Element and in response to the `SSDP:discover` message. Latest after the MirrorLink Client has accessed the Device XML, the MirrorLink Server MUST start the `SSDP:alive` advertisements. Note: In case the MirrorLink Server goes temporarily offline, it MUST send an `SSDP:byebye` message followed by a `SSDP:alive` message, when becoming online again.

UPnP Service Description

The MirrorLink Client MUST follow the Core Specification [8].

MirrorLink Session Setup:

The MirrorLink Client MUST follow the MirrorLink Session Setup sequence defined in the MirrorLink Core Architecture specification [8].

The MirrorLink Client MUST set its Client Profile using UPnP SetClientProfile action over the established Wi-Fi connection.

3.3 Phase 3: WFD Session Setup

Phase 3 MAY start in parallel to the Phase 2, unless the WFD session has been setup outside the MirrorLink session.

Note: WFD TCP connection MUST be established for the purpose of WFD RTSP procedures within the time limit specified in [5]. Similarly, the WFD session setup MUST start after the establishment of TCP connection to ensure that the WFD RTSP timeout requirements in [5] are met.

After successful exchange of RTSP M7 request/response, as shown in Figure 5, the WFD Sink MAY immediately send a RTSP M9 request (PAUSE) to pause the streaming of audio and/or video content from the WFD Source to the WFD Sink. The application launch is typically triggered from the user. The WFD Sink MUST send the M7 request (PLAY) to resume A/V streaming once the 1st application over WFD is launched if AV streaming has been previously paused. Phase 3 MUST NOT be executed, if the WFD session is already setup, e.g. triggered from a previous UPnP Application Launch action.

Testing Considerations:

MirrorLink devices MAY refuse to establish a WFD session to devices not capable of supporting MirrorLink over WFD. In order to pass Miracast certification, those devices MAY implement a specific test mode in which a WFD session with non-MirrorLink WFD devices is possible, which may be disabled afterwards for MirrorLink operation.

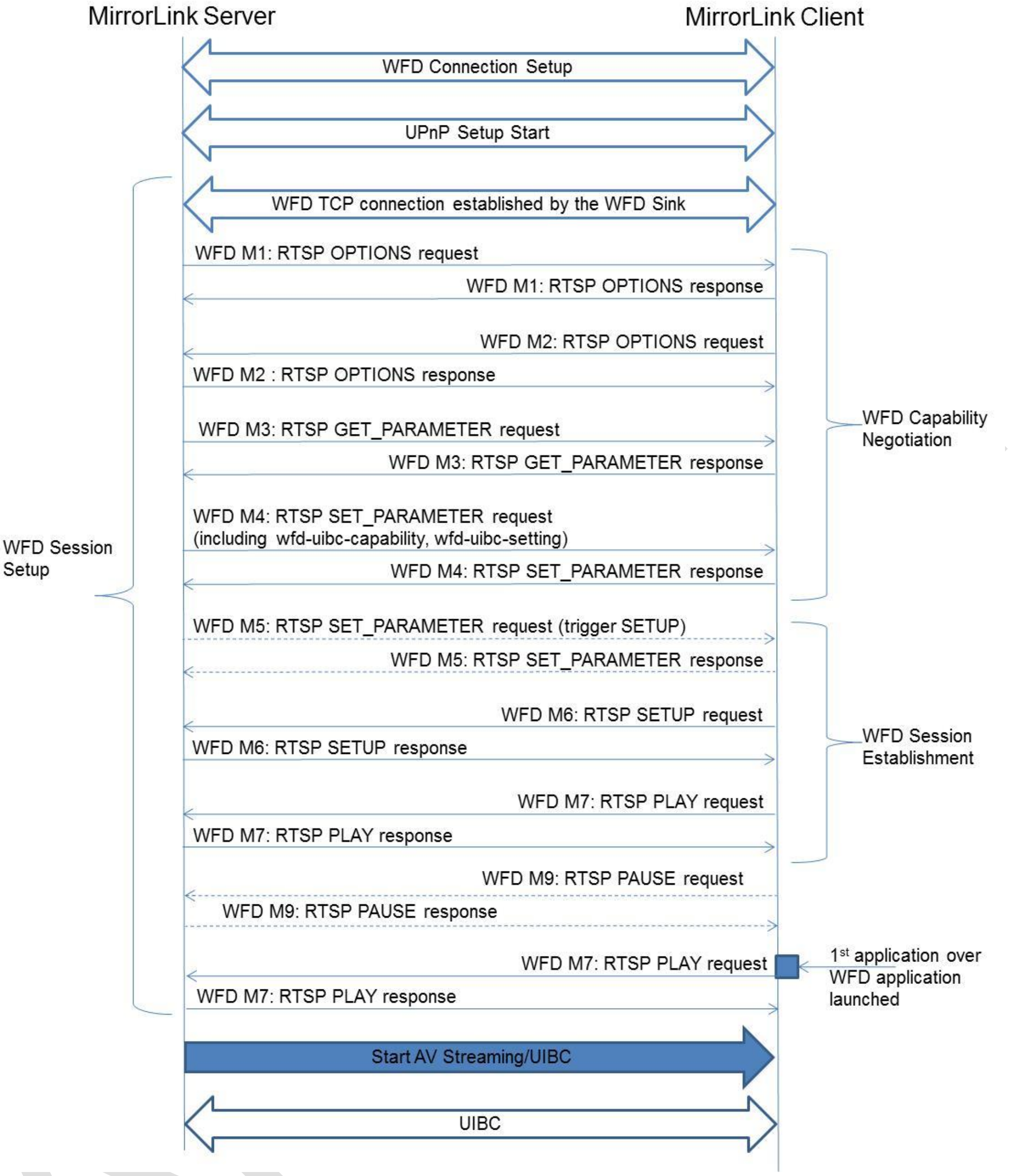
MirrorLink Server
MirrorLink Client
WFD Connection Setup
UPnP Setup Start
WFD TCP connection established by the WFD Sink
WFD M1: RTSP OPTIONS request
WFD M1: RTSP OPTIONS response
WFD M2: RTSP OPTIONS request
WFD M2 : RTSP OPTIONS response
WFD M3: RTSP GET_PARAMETER request
WFD M3: RTSP GET_PARAMETER response
WFD M4: RTSP SET_PARAMETER request
(including wfd-uibc-capability, wfd-uibc-setting)
WFD M4: RTSP SET_PARAMETER response
WFD Capability Negotiation
WFD Session Setup
WFD M5: RTSP SET_PARAMETER request (trigger SETUP)
WFD M5: RTSP SET_PARAMETER response
WFD M6: RTSP SETUP request
WFD M6: RTSP SETUP response
WFD M7: RTSP PLAY request
WFD M7: RTSP PLAY response
WFD Session Establishment
WFD M9: RTSP PAUSE request
WFD M9: RTSP PAUSE response
WFD M7: RTSP PLAY request
WFD M7: RTSP PLAY response
1st application over WFD application launched
Start AV Streaming/UIBC
UIBC

Figure 5: WFD Session Setup Procedure

The following is assumed before A/V streaming in a WFD Session can start:

- WFD connection has been established between the MirrorLink Server and MirrorLink Client.
- UPnP Session has been established.
- MirrorLink Server device has been attested.
- WFD session has been setup.
- UIBC has been established and enabled.

The A/V streaming in WFD session MUST be started only after the launch of the first application via UPnP, which has a protocol identifier (protocolID) value of “WFD”[1]. The WFD Session Setup MUST follow the process of WFD Display Specification as described in [5], and consist of a number of M1, M2, M3, M4, M5, M6 and M7 messages exchanges.

[1] There is no specific “WFD application”, which is advertised separately via UPnP and which has to be launched prior being able to use any user application over WFD.

NOTE

1. For further information of the messages flow of WFD Session Setup, sections 4.6 and 4.8 in [5] can give detail process and description.
2. For further information of the WFD messages used in messages flow, section 6 in [5] can give detail information of format and description.

For the specific UIBC for the car usage, M3/M4 messages MUST contain wfd-uibc-capability parameter. The wfd-uibc-setting parameter MUST be included in the first RTSP M4 request message during the WFD Capability Negotiation.

Note: WFD Source MUST NOT set any capability in the RTSP M4 Request which are not indicated to be supported by the WFD Sink in its RTSP M3 Response.

Once the WFD session has been successfully established with the UIBC enabled, the MirrorLink Server or MirrorLink Client MUST NOT send any M15 Request (RTSP SET_PARAMETER) message to disable the UIBC.

The MirrorLink Server starts Audio/Video streaming to the MirrorLink Client after receiving the RTSP PLAY message from the MirrorLink Client.

The MirrorLink Server and Client MUST support 800x480p30 as the baseline configuration. If a MirrorLink Server and Client device has display resolution of 1280x720 or above, and it is providing WFD functionality, it MUST support 1280x720p30 as well. If the ML Server and Client both support 1280x720p30 resolution, the Server MUST select this display resolution for rendering.

Applications MUST render using the highest resolution supported by the MirrorLink Client and Server as determined during the WFD capability negotiation. They MUST preserve the aspect ratio of the negotiated resolution, while not clipping. The MirrorLink Server MUST add padding if required. The MirrorLink Server MUST NOT stretch its framebuffer to compensate for any difference in the framebuffer aspect ratio.

Note: The ML Server MUST NOT send the display content if the application does not support landscape and only launches in Portrait orientation when the ML Client is in drive mode. When the ML Client is in park mode, the ML Server MAY transmit the framebuffer in Landscape even if the application launches in Portrait orientation.

3.4 Phase 4: WFD Operation

Phase 2 and 3 MUST be completed, before phase 4 can start.

During the WFD Operation phase, the WFD session is controlled from the MirrorLink Client, using RTSP commands and UIBC events. This allows the MirrorLink Client to start or pause the streaming of the MirrorLink Server’s framebuffer. In case the MirrorLink Client does not need the WFD session anymore, it can tear-down the session.

The MirrorLink Client is capable of controlling the MirrorLink Server Audio/Video streaming.

Pause Video, while Audio Continues

The sequence to pause the RTP video streaming, while audio is still being played is shown in

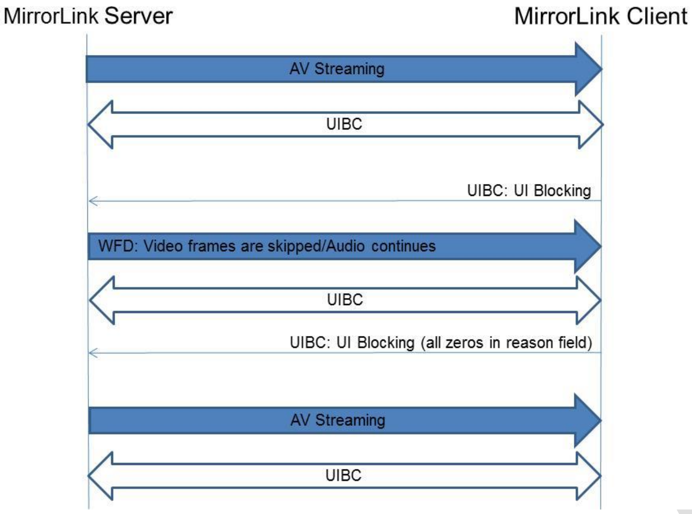
MirrorLink Server
MirrorLink Client
AV Streaming
UIBC
UIBC: UI Blocking
WFD: Video frames are skipped/Audio continues
UIBC
UIBC: UI Blocking (all zeros in reason field)
AV Streaming
UIBC

Figure 6: WFD Operation – Video-only Pause Sequence Diagram

In this case, WFD capability needs to negotiate for video frame skipping to be supported with video frame skipping interval set to infinite. Refer to Wi-Fi Display Specification section 4.10.3.1 for details about video frame skipping feature in WFD.

The MirrorLink Client MUST send a UI blocking message over the WFD UIBC channel with the reason flag "`UI not visible on remote display`" enabled, when it intends to pause the RTP video streaming, while audio is still being needed.

The MirrorLink Client MUST support the video frame skipping feature, specified in Wi-Fi Display Specification section 4.10.3.1. The MirrorLink Server MAY use the video frame skipping feature to start skipping video frames. Audio will continue to be streamed.

The MirrorLink Client MUST send a UI blocking message with all reason flags set to zeros over WFD UIBC channel, if it intends to resume the RTP video streaming. The MirrorLink Server MUST stop using the video frame skipping feature and continue to stream audio and video at the negotiated rates, if the MirrorLink Server had enabled framebuffer skipping.

Pause Audio/Video

The sequence to pause the RTP Audio/Video streaming is shown in Figure 7.

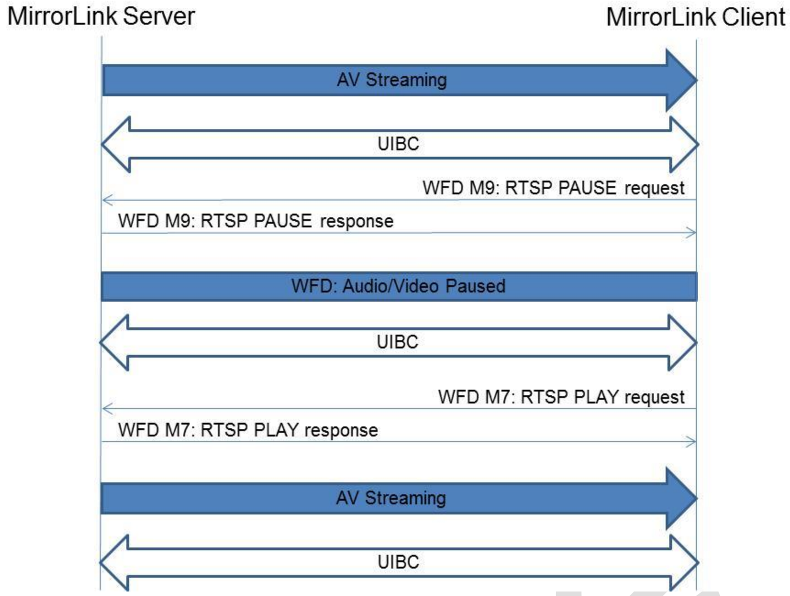
MirrorLink Server
MirrorLink Client
AV Streaming
UIBC
WFD M9: RTSP PAUSE request
WFD M9: RTSP PAUSE response
WFD: Audio/Video Paused
UIBC
WFD M7: RTSP PLAY request
WFD M7: RTSP PLAY response
AV Streaming
UIBC

Figure 7: WFD Operation – Audio/Video Pause Sequence Diagram

The MirrorLink Client MUST send a WFD M9 message (RTSP PAUSE) if it intends to pause the RTP audio/video stream. The MirrorLink Server MUST acknowledge the RTSP PAUSE message and stop AV streaming.

The MirrorLink Client MUST send a WFD M7 message (RTSP PLAY), if it intends to resume the RTP audio/video stream. The MirrorLink Server MUST acknowledge the RTSP PLAY message and resume AV streaming.

Teardown

The MirrorLink Client MUST NOT issue an RTSP TEARDOWN command, unless the MirrorLink session ends or the MirrorLink Client is switching to another remote UI mechanisms.

Otherwise, the MirrorLink Server/Client can reconfigure the WFD session within the limits of the WFD specification.

4 WFD AUDIO BACK CHANNEL

WFD does not provide an Audio Back Channel.

In case the MirrorLink Client intends to setup an Audio Back Channel, it MUST follow the regular MirrorLink mechanism, as specified in the Audio Specification [2]. The Audio Back Channel SHOULD be established before the first WFD Application is launched from the MirrorLink Client.

5 WFD USER INPUT

MirrorLink Server and MirrorLink Client MUST support User Input Back Channel as a mandatory feature of MirrorLink, through which the user input from user interface displayed at MirrorLink Client could be communicated back to MirrorLink Server. In addition user interface related output and status events are exchanged through the User Input Back Channel.

The User Input Back Channel will also be used for certain messages in the forward direction (MirrorLink Server/WFD Source to MirrorLink Client/WFD Sink). Traditionally, the UIBC channel is used for messages going from the WFD Sink to the WFD Source but in MirrorLink, this channel will be used for messages going in both directions.

Messages from MirrorLink Client to MirrorLink Server include the following types:

- Key event;
- Pointer event;
- Touch event;
- Sink Status event;
- UI Blocking event;
- Audio Blocking event;
- Sink Cut Text event;

Messages from MirrorLink Server to MirrorLink Client include the following types:

- Source Status event;
- UI Context event;
- Audio Context event;
- Source Cut Text event;
- Text Output event.

All UIBC user input & output MUST be supported by using an Input Category set to a value 15 (defined in [5] as a Reserved input category, the Generic and HIDC categories may not be supported in MirrorLink setup). The Input Category value 15 is used as a Vendor-Specific category for MirrorLink. The Vendor Specific input body format for MirrorLink is described in Section 5.1.

5.1 UIBC Input Body Format for MirrorLink

The payload structure for packetizing UIBC user inputs are specified in Section 4.11.1 (UIBC Data Encapsulation) of [5].

Note: The UIBC input body field SHOULD be padded up to an integer multiple of 16 bits to have an even integer number in the Length field as recommended in the WFD specification [5].

All MirrorLink specific UIBC inputs shall use the Input Category field set to Vendor-Specific input category (value set to 15), and the format for the UIBC Input Body field is as shown in Table 1 below:

Field	Size (Octet)	Value
OUI	3	04-DF-69 (CCC specific OUI)
Type ID	1	User Input type event ID as listed in Table 2
Length	2	Length of the following fields in octets
Descriptor	Variable	The details of the user inputs

Table 1: UIBC Input Message Format for MirrorLink

Type ID	Notes	Origin	wfd-uibc-capability parameter	Obligation
0	Key Event	Sink	ccc-keys-info	MANDATORY for ML Server OPTIONAL for ML Client
1	Pointer Event	Sink	ccc-pointer-info	OPTIONAL
2	Touch Event	Sink	ccc-touch-info	MANDATORY for ML Server OPTIONAL for ML Client
3	Sink Status Event	Sink	None	MANDATORY
4	Source Status Event	Source	None	MANDATORY
5	UI Context Event	Source	None	MANDATORY
6	UI Blocking Event	Sink	None	MANDATORY
7	Audio Context Event	Source	None	MANDATORY
8	Audio Blocking Event	Sink	None	MANDATORY
9	Sink Cut Text Event	Sink	ccc-inp-type	OPTIONAL
10	Source Cut Text Event	Source	ccc-inp-type	OPTIONAL
11	Text Output Event	Source	ccc-text-output	OPTIONAL
12-255	Reserved			

Table 2: Type IDs for Vendor-Specific UIBC Category for MirrorLink

5.1.1 Key Event

The User Input Descriptor field of the MirrorLink UIBC Input message for the Key Event Input Type ID is shown in Table 3.

Field	Size (Octet)	Notes
Down Flag	1	Non-zero (true) if the key is now pressed, zero (false) if it is now released.
Key Symbol Value	4	Key Symbol Value

Table 3: User Input Descriptor Field for Key Events

The Key Symbol Values are specified in the RFB specification [9] and in Appendix A. Key Events MUST be handled similar to the VNC behavior expressed in [1].

5.1.2 Pointer Event

The User Input Descriptor field of the MirrorLink UIBC Input message for the Pointer Event Input Type ID is shown in Table 4. The pointer event indicates either pointer movement or a pointer button press or release.

Field	Size (Octet)	Notes
Button Mask	1	Current states of buttons 1 to 8 are represented by bits 0 to 7 of the *button-mask* respectively. Each bit value set to 0 means up and 1 means down (pressed).
X-position	2	X-coordinate of the pointer
Y-position	2	Y-coordinate for the pointer

Table 4: User Input Descriptor Field for Pointer Events

The pointer input event MUST NOT be used for single touch events. Single touch events MUST be send as touch event input with one single touch event (N = 0x01) as defined below.

5.1.3 Touch Event

User Input Descriptor field of the MirrorLink UIBC Input message for the Touch Event Input Type ID is shown in Table 5. Touch events are used to describe touch screen action in which the user touches the screen with multiple individual fingers at different locations. The coordinate origin (0, 0) is defined to be the top-left corner of the rectangular display region.

Field	Size (Octet)	Notes
Number of events (N)	1	Number of individual events of a multi-location touch event. When set to 0x01, it indicates a single touch event.
For i = 1: N {	N x 6	
Individual touch event	6	Individual event, consisting of a (x,y) coordinate, an event identifier and a pressure value. The format of the individual event is specified in the Touch Event chapter of [1]
}		

Table 5: User Input Descriptor Field for Touch Events

The WFD sink MUST only use event identifier within the range `[0; Nmax-1]`, where `Nmax` is the maximum number of simultaneous supported touch events, as exchanged within the UIBC ccc-touch-num-info parameter as described in Section 5.2.1. Each event MUST be completed, i.e. each press event MUST be later followed by a release event.

Note: The UIBC header has a time stamp (2 byte field), which SHOULD be used as a time reference for gesture recognition.

The MirrorLink Client MUST provide the coordinates within the framebuffer resolution of the current WFD session.

5.1.4 Sink & Source Status Events

The User Input Descriptor field of the MirrorLink UIBC Input message for the Sink & Source Status Event Type ID is shown in Table 6. The Sink & Source Status event provides status information of specific device features and the ability to set them.

Field	Size (Octet)	Notes
Status	4	Status of device features as specified in the Device Status Messages chapter of [1].

Table 6: User Input Descriptor Field for Sink & Source Status Events

5.1.5 UI Context Event

The User Input Descriptor field of the MirrorLink UIBC Input message for the UI Context Event Type ID is shown in Table 7.

Field	Size (Octet)	Notes
Reference	4	Context reference within the RTP UI stream. This Reference field MUST be set to a non-zero value; the value 0 is reserved.
Number (N)	1	Number of UI context information.
For i = 1: N {	N x 24	
X	2	X-position of rectangle (top left corner)
Y	2	Y-position of rectangle (top left corner)

Field	Size (Octet)	Notes
Width	2	Width of rectangle
Height	2	Height of rectangle
App ID	4	Unique application identifier, the UI is originating from. Applications being advertised via UPnP, MUST match the advertised AppID; otherwise set to zero.
Trust level	2	Trust Level for Application category (see [7] Table 6-1).
Trust level	2	Trust Level for Content category (see [7] Table 6-1).
Application category	4	Application category (see [7] Table 6-2).
Content category	4	Content category (see [7] Table 6-3).
}		

Table 7: User Input Descriptor Field for UI Context Events

The WFD Source MUST provide the initial UI Context Events with the start of the RTP streaming. The WFD Sink SHOULD NOT show any content, prior the initial UI Context Event has been received. The WFD Source MUST provide context information, whenever there is a change in the context information,

If the WFD Sink receives a UI Context event, with an application category set to "`Switch to MirrorLink Client native UI`" (`0xF000FFFF`) the MirrorLink Client MUST switch to a native user-interface.

The WFD Source MUST provide context information for the entire display, i.e. the aggregation of the individual rectangular areas MUST always cover the entire framebuffer, and never a partial framebuffer area alone. If multiple overlapping rectangles are given, the sequence of the rectangles defines the stacking order (last rectangle on top).

5.1.6 UI Blocking Event

The User Input Descriptor field of the MirrorLink UIBC Input message for the UI Blocking Event Type ID is shown in Table 8.

Field	Size (Octet)	Notes
Reference	4	Context reference to the RTP UI stream. The UI blocking reference MUST be equal to the UI context event's reference, which is subject to block. MUST be zero if blocking is not related to any UI context event.
Number (M)	1	Number of UI blocking information.
For i = 1: M {		
Index	1	UI context information index (N) starting at one, which is subject to block. The Index MUST be zero if blocking is not related to any UI context information.
Reason	2	Reason for blocking UI (bitmask) The bit mask is specified in the "Reason for Blocking" entry in the Framebuffer Blocking Notification Message of [1]. All-zero reason flags in this field indicates that the referenced RTP UI stream MUST be unblocked.
}		

Table 8: User Input Descriptor Field for UI Blocking Events

The WFD Sink MAY send an UI Blocking event, with the reason flags being all-zeros, to indicate that the RTP UI stream MUST be unblocked.

The Reference field is used to uniquely identify a UI Context event sent from the Source to the Sink. In order to block the framebuffer of an application, the WFD Sink should send a UI Blocking event using the same Reference field and UI context information index.

Details on how the MirrorLink Server and Client MUST handle the blocking of the User Interface are defined in the VNC specification, section 5.9 [1].

5.1.7 Audio Context Event

The User Input Descriptor field of the MirrorLink UIBC Input message for the Audio Context Event Type ID is shown in Table 9.

Field	Size (Octet)	Notes
Reference	4	Context reference to the RTP audio stream. This Reference field MUST be set to a non-zero value; the value 0 is reserved.
Number (N)	1	Number of audio context information. The audio context information number MUST be set to zero (0), if the RTP stream does not carry real audio anymore.
For i = 1: N {	N x 8	Note: No list, if N equals zero (0).
App ID	4	Unique application identifier, the audio is originating from. Applications being advertised via UPnP, MUST match the advertised AppID; otherwise set to zero.
Application category	4	Application category (see [7] Table 6-2).
}		

Table 9: User Input Descriptor Field for Audio Context Events

Setting the audio context information number to zero, is meant as an indication from the MirrorLink Server (and its applications), that intentionally no further audio is going to be provided at the moment. The MirrorLink Server MAY resume the audio playback at any later time though.

The MirrorLink Server MUST send an Audio Context Event, when the value of Number (N) changes.

The MirrorLink Client SHOULD use the Number of audio context information as a trigger to:

- Fade In/Out to local audio sources, if N becomes zero and audio is available from the MirrorLink Client.
- Fade In/Out to MirrorLink audio sources, if N becomes non-zero and no other higher-priority audio is available from the MirrorLink Client.

5.1.8 Audio Blocking Event

The User Input Descriptor field of the MirrorLink UIBC Input message for the Audio Blocking Event Type ID is shown in Table 10.

Field	Size (Octet)	Notes
Reference	4	Context reference within the RTP audio stream. The audio blocking reference MUST be equal to the audio context event’s reference, which is subject to block. MUST be zero if blocking is not related to any audio context event.

Number (M)	1	Number of audio blocking information.
For i = 1: M {		
Index	1	Audio context information index (N) starting at one, which is subject to block. The Index MUST be zero if blocking is not related to any audio context information.
Reason	2	Reason for blocking audio (bitmask) The bitmask is specified in the "Reason for blocking" entry in the Audio Blocking Notification Message of [1].
}		

Table 10: User Input Descriptor Field for Audio Blocking Events

The Reference field is used to uniquely identify an audio Context event sent from the Source to the Sink. In order to block the audio of an application, the WFD Sink MUST send an audio blocking event using the same Reference field and the Audio context information index.

Details on how the MirrorLink Server and Client MUST handle the blocking of the Audio are defined in the VNC specification, section 5.10 [1].

5.1.9 Sink& Source Cut Text Events

The User Input Descriptor field of the MirrorLink UIBC Input message for the Sink & Source Cut Text Event Input Type ID is shown in Table 11. Ends of lines are represented by the linefeed / newline character (value 0xFF0A) alone. No carriage-return (value 0xFF0D) is needed.

Field	Size (Octet)	Notes
Length	4	Number of UTF16 characters of the text content
Text	*Length*	Text content as an array of UTF16 characters of the specified length

Table 11: User Input Descriptor Field for Sink & Source Cut Text Events

5.1.10 Text Output Event

The User Input Descriptor field of the MirrorLink UIBC Input message for the Text Output Event Type ID is shown in Table 12.

Field	Size (Octet)	Notes
App ID	4	Unique application id. Applications being advertised via UPnP, MUST match the advertised AppID.
Length	4	Number of UTF16 characters of the text content
Text	*Length*	Text content as an array of UTF16 characters of the specified length

Table 12: User Input Descriptor Field for Text Output Events

The provided textual-information is valid for the identified application until it is either overridden from a new message or invalidated (i.e. the length of the textual information is zero). Multiple valid textual-information entries can exist in parallel for different uniquely identifiable applications.

5.2 MirrorLink Specific RTSP Data Structures

The UIBC capability negotiation and update process are achieved using RTSP procedures.

MirrorLink Client and Servers MUST use UIBC (User Input Back Channel) to implement the User Input & Output Event mechanisms. During RTSP capability negotiation phase, wfd-uibc-capability parameter is used to describe what UIBC related attributes are supported, and the wfd-uibc-setting parameter is used to enable the UIBC session. The MirrorLink specific event inputs are included as the vendor-specific-cap-info parameter within the wfd-uibc-capability.

5.2.1 wfd-uibc-capability parameter

When using WFD for MirrorLink, the wfd-uibc-capability parameter (as described in section 6.1.15 of [5]) is used with input-cat field set to “VENDOR_SPECIFIC” and the vendor-specific-cap-info field set as specified (in the bold-face) as below.

```
    wfd-uibc-capability          = “wfd_uibc_capability:” SP (“none” / (input-
                                   category-val “;” generic-cap-val “;” hidc-
                                   cap-val “;” vendor-specific-cap-info “;”
                                   tcp-port)) CRLF; “none” if not supported
    input-category-val           = “input_category_list=” (“none” / input-
                                   category-list)
    input-category-list          = input-cat * (“,” SP input-category-list)
    input-cat                    = “GENERIC” / “HIDC” / ”VENDOR_SPECIFIC”
    generic-cap-val              = “generic_cap_list=” (“none” / generic-cap-
                                   list)
    generic-cap-list             = inp-type *(“,” SP generic-cap-list)
    inp-type                     = “Keyboard” / “Mouse” / “SingleTouch” /
                                   “MultiTouch” / “Joystick” / “Camera” /
                                   “Gesture” / “RemoteControl”
    hidc-cap-val                 = “hidc_cap_list=” (“none” / hidc-cap-list)
    hidc-cap-list                = detailed-cap *(“,” SP hidc-cap-list)
    detailed-cap                 = inp-type “/” inp-path
    inp-path                     = “Infrared” / “USB” / “BT” / “Zigbee” / “Wi-
                                   Fi” / “No-SP”
    vendor-specific-cap-info     = “vendor_specific_cap_info=” (“none” /
                                   (“OUI:“ SP vendor-OUI “;” vendor-specific-
                                   cap-list))
    vendor-OUI                   = 6*6HEXDIG; value set to “04DF69” for CCC OUI
                                   when used in MirrorLink setup
    vendor-specific-cap-list     = ccc-event-cap-val; list to include one or
                                   more vendor specific UIBC related capability
                                   parameters
    ccc-event-cap-val            = “ccc_event_cap_list=” (“none” / ccc-event-
                                   inp-type)
    ccc-event-inp-type           = ccc-resolution-info SP ccc-display-info SP
                                   ccc-keys-info SP ccc-pointer-info SP ccc-
                                   touch-info SP ccc-text-output SP ccc-cut-
                                   text
    ccc-resolution-info          = “Resolution/” ccc-resolution-width-info SP
                                   ccc-resolution-height-info
    ccc-resolution-width-info    = 4*4HEXDIG; Sink display resolution width
                                   [pixel]
    ccc-resolution-height-info   = 4*4HEXDIG; Sink display resolution height
                                   [pixel]
    ccc-display-info             = “Display/” ccc-display-width-info SP ccc-
                                   display-height-info SP ccc-display-distance-
                                   info
    ccc-display-width-info       = 4*4HEXDIG; Sink display width [mm]
    ccc-display-height-info      = 4*4HEXDIG; Sink display height [mm]
    ccc-display-distance-info    = 4*4HEXDIG; Sink display distance to user
                                   [mm]
    ccc-keys-info              = ccc-keys-knob-info SP ccc-keys-device-info
                                 SP ccc-keys-multimedia-info SP ccc-keys-
                                 function-info SP ccc-keys-itu-info
    ccc-keys-knob-info         = “KnobKeys/” 8*8HEXDIG; knob keys (Bit mask
                                 according to [1] table 41)
    ccc-keys-device-info       = “DeviceKeys/” 8*8HEXDIG; device keys (Bit
                                 mask according to [1] table 43)
    ccc-keys-multimedia-info   = “MultimediaKeys/” 8*8HEXDIG; multimedia keys
                                 (Bit mask according to [1] table 43)
    ccc-keys-function-info     = “FunctionKeys/” 2*2HEXDIG; number of
                                 additional function keys
    ccc-keys-itu-info          = “ITUKeys/” (“none” / “Supported”); none if
                                 not supported
    ccc-pointer-info           = “Pointer/” 2*2HEXDIG; pointer event button
                                 mask (according to [1])
    ccc-touch-info             = “Touch/” ccc-touch-num-info SP ccc-touch-
                                 pressure-info
    ccc-touch-num-info         = 2*2HEXDIG; number of supported simultaneous
                                 touch events -1
    ccc-touch-pressure-info    = 2*2HEXDIG; touch event pressure mask
    ccc-text-output            = “TextOutput/” 4*4HEXDIG; maximum length of
                                 textual meta information
    ccc-cut-text               = “CutText/” (“none” / “Supported”); none if
                                 not supported
    tcp-port                   = “port=” (“none” / IPPORT)
```

All other fields of wfd-uibc-capability parameters are set as specified in [5].

When the MirrorLink Client does not support some of the keys or events included in the ccc-event-cap-val field of the wfd-uibc-capability parameter, the field values that are not supported MUST be set as per the following table.

ccc-event-cap-val sub-parameters in wfd-uibc-capability	Fields included by the ML Client	Field value to be set if not Supported by the ML Client
ccc-event-inp-type	ccc-resolution-info	N/A (must be set to non-zero pixel values)
	ccc-display-info	N/A (must set to non-zero values for width and height. Distance may be set to 0 if unknown)
	ccc-keys-info	Setting for sub-parameters included below
	ccc-pointer-info	Pointer/0x00
	ccc-touch-info	Touch/0x00 0x00
	ccc-text-output	TextOutput/0x0000
	ccc-cut-text	CutText/none
ccc-keys-info:	ccc-keys-knob-info	KnobKeys/0x00000000
	ccc-keys-device-info	DeviceKeys/0x00000000
	ccc-keys-multimedia-info	MultimediaKeys/0x00000000
	ccc-keys-function-info	FunctionKeys/0x00

ccc-event-cap-val sub-parameters in wfd-uibc-capability	Fields included by the ML Client	Field value to be set if not Supported by the ML Client
	ccc-keys-itu-info	ITUKeys/none

Table 13: Setting of ccc-event-cap-val field in the wfd-uibc-capability parameter

Note: Capability setting of the above events MUST be identical to the capability setting for the VNC session as defined in the VNC specification [1].

5.2.2 wfd-uibc-setting parameter

When using WFD for MirrorLink, the wfd-uibc-setting parameter (as described in section 6.1.16 of [5]) SHALL be used during WFD capability negotiation or with the M15 RTSP message with uibc-setting field set to “enable” to start the UIBC session.

6 WFD CONTENT PROTECTION

Content protection using HDCP MAY be used by following the procedures described in [5]. Additional technical guidance and recommended best practices for MirrorLink devices implementing WFD with HDCP is provided in [10]. When using ML over WFD, the ML server MUST NOT require HDCP content protection to start streaming of A/V content not requiring protection.

The WFD Source MUST inject a message into the video stream, showing an Error message to the user, if no HDCP session can be established, as described in section 3.3.3 of [10].

7 REFERENCES

[1] Car Connectivity Consortium, “MirrorLink – VNC Based Display and Control”, Version 1.1, CCC-TS-010.

[2] Car Connectivity Consortium, “MirrorLink – Audio”, Version 1.1, CCC-TS-012

[3] Car Connectivity Consortium, “MirrorLink – IEEE 802.11 CCC Information Element”, Version 1.2, CCC-TS-050

[4] Car Connectivity Consortium, “MirrorLink – UPnP Server Device”, Version 1.2, CCC-TS-062

[5] Wi-Fi Display Technical Specification Version 1.0.0, September 2012

[6] IETF, RFC 2119, Keys words for use in RFCs to Indicate Requirement Levels, March 1997. 21 http://www.ietf.org/rfc/rfc2119.

[7] Car Connectivity Consortium, “MirrorLink - UPnP Application Server Service”, Version1.2, CCC-TS-060

[8] Car Connectivity Consortium, “MirrorLink – Core Architecture”, Version 1.2, CCC-TS-063

[9] Tristan Richardson, “The RFB Protocol”, RealVNC Ltd, Version 3.8, August 28, 2008.

[10] Best Practices Document for Miracast™ Devices – HDCP2 Related Implementations, Wi-Fi Alliance, July 2014

8 APPENDIX A – KEY EVENT TABLE

The Key event mapping for different 2D knobs is shown in the following Table. The key event mapping for a particular head-unit knob n MUST be according the following format:

```
0x3000 00nm
```

The value n defines the head-unit knob and m defines the event as defined in the template above. Allowed values for n are [0:3] and for **m** are [0:F].

Category	Mnemonic	KeySymValue	Description
Knob Keys	Knob_2D_n_shift_right	0x3000 00n0	Right shift
	Knob_2D_n_shift_left	0x3000 00n1	Left shift
	Knob_2D _n_shift_up	0x3000 00n2	Up shift
	Knob_2D_n_shift_up_right	0x3000 00n3	Up & right shift
	Knob_2D_n_shift_up_left	0x3000 00n4	Up & left shift
	Knob_2D_n_shift_down	0x3000 00n5	Down shift
	Knob_2D_n_shift_down_right	0x3000 00n6	Down & right shift
	Knob_2D_n_shift_down_left	0x3000 00n7	Down & left shift
	Knob_2D_n_shift_push	0x3000 00n8	Push
	Knob_2D_n_shift_pull	0x3000 00n9	Pull
	Knob_2D_n_rotate_x	0x3000 00nA	x clockwise rotation
	Knob_2D_n_rotate_X	0x3000 00nB	x anti-clockwise rotation
	Knob_2D_n_rotate_y	0x3000 00nC	y clockwise rotation
	Knob_2D_n_rotate_Y	0x3000 00nD	y anti-clockwise rotation
	Knob_2D_n_rotate_z	0x3000 00nE	z clockwise rotation
	Knob_2D_n_rotate_Z	0x3000 00nF	z anti-clockwise rotation
ITU Keys	ITU_Key_0	0x3000 0100	0, ' '
	ITU_Key_1	0x3000 0101	1, '.', ','
	ITU_Key_2	0x3000 0102	2, a, b, c
	ITU_Key_3	0x3000 0103	3, d, e, f
	ITU_Key_4	0x3000 0104	4, g, h, i
	ITU_Key_5	0x3000 0105	5, j, k, l
	ITU_Key_6	0x3000 0106	6, m, n, 0
	ITU_Key_7	0x3000 0107	7, p,q, r, s
	ITU_Key_8	0x3000 0108	8, t, u, v
	ITU_Key_9	0x3000 0109	9, w, x, y, z
	ITU_Key_Asterix	0x3000 010A	*, +
	ITU_Key_Pound	0x3000 010B	#, shift
Device Keys	Device_Phone_call	0x3000 0200	Take a phone call
	Device_Phone_end	0x3000 0201	End phone call
	Device_Soft_left	0x3000 0202	Left soft key

Category	Mnemonic	KeySymValue	Description
	Device_Soft_middle	0x3000 0203	Middle soft key
	Device_Soft_right	0x3000 0204	Right soft key
	Device_Application	0x3000 0205	Shortcut to the Application listing
	Device_Ok	0x3000 0206	Ok
	Device_Delete	0x3000 0207	Delete (Backspace)
	Device_Zoom_in	0x3000 0208	Zoom in
	Device_Zoom_out	0x3000 0209	Zoom out
	Device_Clear	0x3000 020A	Clear
	Device_Forward	0x3000 020B	Go one step forward
	Device_Backward	0x3000 020C	Go one step backward
	Device_Home	0x3000 020D	Shortcut to the Home Screen
	Device_Search	0x3000 020E	Shortcut to the search function
	Device_Menu	0x3000 020F	Shortcut to the (application) menu
Function Keys	Function_Key_0	0x3000 0300	Soft and hard keys available on the client and server user interface
	Function_Key_1	0x3000 0301	
	…	…	
	Function_Key_254	0x3000 03FE	
	Function_Key_255	0x3000 03FF	Reserved
Multimedia Keys	Multimedia_Play	0x3000 0400	Start media playing
	Multimedia_Pause	0x3000 0401	Pause media playing
	Multimedia_Stop	0x3000 0402	Stop media playing
	Multimedia_Forward	0x3000 0403	Forward
	Multimedia_Rewind	0x3000 0404	Rewind
	Multimedia_Next	0x3000 0405	Go to next track in playlist
	Multimedia_Previous	0x3000 0406	Go to previous track in playlist
	Multimedia_Mute	0x3000 0407	Mute the audio stream at source
	Multimedia_Unmute	0x3000 0408	Unmute the audio stream at source
	Multimedia_Photo	0x3000 0409	Take a photo