ETSI TS 103 544-9 V1.3.1 (2019-10)

Publicly Available Specification (PAS); Intelligent Transport Systems (ITS); MirrorLink®; Part 9: UPnP Application Server Service

CAUTION

The present document has been submitted to ETSI as a PAS produced by CCC and approved by the ETSI Technical Committee Intelligent Transport Systems (ITS).

Reference
RTS/ITS-98-9

Keywords
interface, ITS, PAS, smartphone

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status.
Information on the current status of this and other ETSI documents is available at
https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Intelligent Transport Systems (ITS).

The present document is part 9 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.1].

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document is part of the MirrorLink® specification which specifies an interface for enabling remote user interaction of a mobile device via another device. The present document 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 *TmApplicationServer* service is a UPnP service that allows UPnP Control Points to remotely launch and terminate applications on MirrorLink Server devices. Through this service, UPnP control points can provide more fine-grained control and access to specific remote applications.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

[1] UPnP™ Forum: "UPnP™ Device Architecture 1.1", 15 October 2008.

NOTE: Available at http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf.

[2] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax", January 2005.

NOTE: Available at http://tools.ietf.org/html/rfc3986.

[3] W3C Recommendation 11 April 2013: "XML Signature Syntax and Processing Version 1.1".

NOTE: Available at http://www.w3.org/TR/xmldsig-core/.

[4] ETSI TS 103 544-26 (V1.3.1): "Publicly Available Specification (PAS); Intelligent Transport Systems (ITS); MirrorLink® ; Part 26: Consumer Experience Principles and Basic Features".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 103 544-1 (V1.3.1): "Publicly Available Specification (PAS); Intelligent Transport Systems (ITS); MirrorLink®; Part 1: Connectivity".

3 Definition of terms, symbols and abbreviations

3.1 Terms

Void.

3.2 Symbols

Void.

3.3 Abbreviations

Void.

4 Service Modeling Definitions

4.1 Service Type

The following service type identifies a service that is compliant with the present document:

- **urn:schemas-upnp-org:service**:*TmApplicationServer:1*.

TmApplicationServer service is used herein to refer to this service type. The *TmApplicationServer* service shall follow defined UPnP behaviour within the UPnP Device Architecture 1.1 [1].

4.2 State Variables

4.2.1 State Variable Overview

Table 4-1: Service State Variables

Variable Name	Req. or Opt.	Data Type	Allowed Value	Default Value	Eng. Units
AppStatusUpdate	R	string	Undefined	Empty string	N/A
AppListUpdate	R	string	Undefined	Empty string	N/A
A_ARG_TYPE_AppStatus	R	string	Undefined	Empty string	N/A
A_ARG_TYPE_AppID	R	string	Undefined	Empty string	N/A
A_ARG_TYPE_ProfileID	R	ui4	Undefined	0	N/A
A_ARG_TYPE_URI	R	string	Undefined	Empty string	N/A
A_ARG_TYPE_AppList	R	string	Undefined	Empty string	N/A
A_ARG_TYPE_String	R	string	Undefined	Empty string	N/A
A_ARG_TYPE_Bool	R	string	true | false	false	N/A
A_ARG_TYPE_INT	R	ui4	Undefined	0	N/A
A_ARG_TYPE_AppCertificateInfo	R	string	Undefined	Empty string	N/A
R = REQUIRED. *O* = OPTIONAL. *X* = Non-standard.					

4.2.2 AppStatusUpdate

A string formatted as UTF-8 represents the list of application identifiers (*appIDs*) of applications whose status has changed. The string consists of a comma-separated list of appIDs identifying applications whose status has changed. Each entry in the list is of the type *A_ARG_TYPE_AppID*.

This state variable is evented, implying that clients can subscribe to receive notifications every time the variable changes using UPnP standardized eventing mechanisms. It is important to note that this variable only contains the *appIDs* of those applications, whose status has changed since the last time an event notification was sent out.

On receiving an *AppStatusUpdate* event, the MirrorLink UPnP Control Point can query the application status of specific applications in the list by invoking the *GetApplicationStatus* action.

AppStatusUpdate value will consist of a comma separated list of all application identifiers (*appIDs*) of applications listed in *A_ARG_TYPE_AppList* when the event is issued by the *TmApplicationServer* service for the first time.

After an application launch, the MirrorLink Server shall only send the *AppStatusUpdate* event, once the application is running and in foreground. The *AppStatusUpdate* shall be send only after the response to the UPnP *LaunchApplication* or *TerminateApplication* action has been sent for UI applications.

The MirrorLink Server will provide information of the current foreground framebuffer also via the VNC/WFD context information. During an application launch or termination action, this information can be temporarily out of sync with the UPnP application status, e.g. the UPnP information might be trailing the VNC context info in case of an application launch. In case the framebuffer transfers have been paused, e.g. due to the MirrorLink application being in the background on the MirrorLink Client screen, the VNC context information will not be updated until the framebuffer transfer is resumed.

4.2.3 AppListUpdate

A string formatted as UTF-8 represents a list of application identifiers (*appIDs*) of applications whose entries have changed in the application listing. The string consists of a comma-separated list of *appIDs* identifying applications whose status has changed. Each entry in the list is of the type *A_ARG_TYPE_AppID*.

It is evented, implying that clients can subscribe to receive notifications every time the variable changes using UPnP standardized eventing mechanisms. It is important to note that this variable only contains the appIDs of those applications, whose entries in the application list have changed since the last time an event notification was sent out.

On receiving an *AppListUpdate* event, a MirrorLink UPnP Control Point can retrieve the application list by invoking the *GetApplicationList* action and specifying the appropriate filter using the *appListingFilter* input argument.

AppListUpdate value will consist of a comma separated list of all application identifiers (*appIDs*) of applications listed in *A_ARG_TYPE_AppList* when the event is issued by the *TmApplicationServer* service for the first time.

The MirrorLink Client shall follow the *AppListUpdate* event. This will ensure that a revocation of an application certificate, specifically for drive-certified applications, will take immediate effect. Additionally, newly installed applications or applications for which an application certificate has been successfully downloaded from the ACMS, are immediately available, without reconnecting the MirrorLink session.

Implementation Note:

> Older MirrorLink 1.1 Clients may either ignore *AppListUpdate* events or do not subscribe to them. Consumers will need to reestablish a MirrorLink session, in order to see a MirrorLink application available on the MirrorLink Client, if it has been installed from within a MirrorLink session.

The MirrorLink Server shall send an *AppListUpdate* event only in case a change to an application entry or to the certification status happened. The MirrorLink Server should combine changes to multiple applications into a single event.

Implementation Note:

> Older MirrorLink 1.1 Servers may send an *AppListUpdate*, even in case nothing has changed in the application list. It is recommended, that MirrorLink Clients check the application listing as they cannot distinguish the MirrorLink Server's behavior.

4.2.4 A_ARG_TYPE_AppStatus

A string formatted as UTF-8 XML represents the status of a specific application or alternatively providing the status of all applications, which can be controlled remotely. Its structure is given in Table 4-2.

Table 4-2: Structure of A_ARG_TYPE_AppStatus

Element	Description	Parent	Availability
appStatusList	Indicates list of application status updates	-	Required
appStatus*	Indicates status record corresponding to an application	appStatusList	Required
appID	Unique ID of the application (A_ARG_TYPE_AppID)	appStatus	Required
status*	Entry corresponding to an instance of the application running under a specific client profile	appStatus	Required
profileID	Profile Identifier of the client profile (A_ARG_TYPE_ProfileID)	status	Required
statusType	String representing status of application: {Foreground | Background | Notrunning} (A_ARG_TYPE_String)	status	Required

The elements marked with a (*) can have multiple instances.

4.2.5 A_ARG_TYPE_AppID

A UTF-8 encoded string represents an unsigned 32-bit integer in hexadecimal format (with '0x' prefix) which denotes the unique application identifier.

The MirrorLink Server shall use the unsigned integer value of a variable of this type within any action. I.e. comparing the values of two A_ARG_TYPE_AppID variables shall be done based on the unsigned integer value and not based on a specific character representation.

Therefore, the following two A_ARG_TYPE_AppID values are identical:

- `0x45ab` and `0x45AB` (case insensitivity of the hexadecimal numbers).
- `0x45ab` and `0X45ab` (case insensitivity of the 0x).
- `0x00001234` and `0x001234` (leading zeros do not matter).

NOTE: The application identifier should be the same over time for the same application (e.g. should survive a reboot or MirrorLink reconnect), to allow the MirrorLink Client to implement a Last-Mode behavior.

An *A_ARG_TYPE_AppID* value may be identical to the wildcard **"*"**, but it shall not be used, unless its usage is specifically stated in the definition of the respective UPnP actions and/or events.

4.2.6 A_ARG_TYPE_ProfileID

An unsigned 32-bit integer greater than or equal to 0, represents a unique profile identifier. Its value is set equal to 0 by default.

4.2.7 A_ARG_TYPE_AppList

A string formatted as UTF-8 XML represents the list of all applications that are available for remote control and access through the TmApplicationServer service. Its structure is given in Table 4-3. Server devices shall be able to support values of *A_ARG_TYPE_AppList* up to 10 KiloBytes in length.

Table 4-3: Structure of A_ARG_TYPE_AppList

Element	Description	Parent	Availability
appList	List of all available remote applications	-	Required
app*	Entry describing one remote application	appList	Optional
appID	Unique application ID. Shall be non-zero (A_ARG_TYPE_AppID)	app	Required
name	Application name (A_ARG_TYPE_String)	app	Required
variant	Unique application ID (*appID*) of the parent application, presented in the UPnP application listing (A_ARG_TYPE_String)	app	Optional
provider Name	Name of the application provider (A_ARG_TYPE_String)	app	Optional
provider URL	URL of the application provider's website (A_ARG_TYPE_URI)	app	Optional
description	Text description of application (A_ARG_TYPE_String)	app	Optional
iconList	List of available application icons. First icon shall be either *mimetype*=`image/png`, *width*=128, *height*=128 and *depth*=24 (default), or identical to values set in the client icon preferences as specified using the *TmClientProfileServer*:1 service's *SetClientProfile* action The MirrorLink Client shall support displaying icons with mimetype=`image/png`, width=128, height=128 and depth=24	app	Optional
icon*	Describes an application icon. The MirrorLink Server shall include an icon for all applications with the following <*protocolID*> values: • VNC • WFD (MirrorLink ≥ 1.2)	iconList	Optional
mimetype	Type of icon image (A_ARG_TYPE_String)	icon	Required
width	Width of icon (A_ARG_TYPE_INT)	icon	Required
height	Height of icon (A_ARG_TYPE_INT)	icon	Required
depth	Color depth of icon (A_ARG_TYPE_INT)	icon	Required
url	URL where icon is available. If the icon of an application changes, the MirrorLink Server should change the icon's url. MirrorLink Client shall use HTTP-GET to access the icon behind the URL. (A_ARG_TYPE_URI)	icon	Required
allowed ProfileIDs	Reserved for future use **Deprecated**	app	Deprecated
remoting Info	Information about the remoting protocol used to interact with the application after it is launched	app	Required
protocolID	Protocol Identifier of the remoting protocol that will be used to access the application (see Table 4-4 for list of supported protocols) (A_ARG_TYPE_String)	remoting Info	Required
format	Format of data being transferred using the remoting protocol (see Table 4-5 for details) (A_ARG_TYPE_String)	remoting Info	Optional

Element	Description	Parent	Availability
direction	Direction of the content stream. A_ARG_TYPE_String with one of the following values: • "out" - Content streaming from the MirrorLink server device to the client • "in" - Content streaming from the MirrorLink client to the server • "bi" - Content streaming in both directions between the MirrorLink server and client **Default:** "out"	remoting Info	Optional
audioIPL	Audio Initial Playback Latency (A_ARG_TYPE_INT) **Default:** "4800"	remoting Info	Optional
audioMPL	Audio Maximum Playback Length (A_ARG_TYPE_INT) **Default:** "9600"	remoting Info	Optional
app Certificate URL	URL where application certificate is available. The MirrorLink Client uses the certificate for information purpose. The MirrorLink Client shall not validate the certificate's trust chain MirrorLink Client shall use HTTP-GET to access the certificate behind the URL (A_ARG_TYPE_URI)	app	Optional
appInfo	Information about the listed application	app	Optional
app Category	Application category (A UTF-8 encoded string representing an unsigned 32-bit integer in hexadecimal format (with '0x' prefix).) Values are defined in Annex A **Default:** "0x00000000"	appInfo	Optional
trustLevel	Trust level of the contents of the appInfo element (A UTF-8 encoded string representing an unsigned 16-bit integer in hexadecimal format (with '0x' prefix).) Values are defined in Annex A **Default:** "0x0000"	appInfo	Optional
displayInfo	Information about display content of the listed application, in case it provides a displayable user interface	app	Optional
content Category	Visual content categories used (A UTF-8 encoded string representing an unsigned 32-bit integer in hexadecimal format (with '0x' prefix).) Values are defined in Annex A. **Default:** "0x00000000"	display Info	Optional
content Rules	Visual content rules followed **Deprecated**	display Info	Deprecated
orientation	Display orientations supported. **Deprecated**	display Info	Deprecated
trustLevel	Trust level of the displayInfo element. Value shall match *appInfo/trustLevel*. **Default**: "0x0000"	display Info	Optional
audioInfo	Information about audio content of the listed application, in case it provides an audio interface	app	Optional(++)

Element	Description	Parent	Availability
audioType	Audio type A_ARG_TYPE_String with one of the following values: • "phone" - Phone call audio • "application" - Generic application audio • "all" - Phone and application audio • "none" - no audio	audioInfo	Required
content Category	Audio content categories used (A UTF-8 encoded string representing an unsigned 32-bit integer in hexadecimal format (with '0x' prefix).) Values are defined in Annex A.	audioInfo	Required
content Rules	Audio content rules followed **Deprecated**	audioInfo	Deprecated
trustLevel	Trust level of the audioInfo element (A UTF-8 encoded string representing an unsigned 16-bit integer in hexadecimal format (with '0x' prefix).) Values are defined in Annex A. **Default**: "0x0000"	audioInfo	Optional
resource Status	Application resource status In case the remote application is using a resource, which is subject to access control (e.g. an audio source or sink), this element will define the current status having one of the following values (A_ARG_TYPE_String): • "free" - Resource is free. Can be used by the MirrorLink client • "busy" - Resource already used. Resource assignment can be overridden by a client's invocation LaunchApplication action • "NA" - Resource already used. Resource assignment cannot be overridden by a LaunchApplication action invoked by a client **Default**: "free"	app	Optional
Signature	XML signature over entire contents of the appList element. This is done as specified in [3]. The key used in calculating the signature shall be the private part of the application-specific key which public part was bound to the attestation of UPnP-Server component. (The public part can be used to verify the signature.) The Reference element of the XML signature shall point to appList element. The SignatureMethod shall be RSA with SHA1. The KeyInfo element may be omitted. The mechanism for generation, exchange and maintainance of keys is out of scope for the present document	appList	Required

The elements marked with a (*) can have multiple instances.

(++) The *audioInfo* element shall be included into the advertisement on any RTP Client, RTP Server, BT HFP or BT A2DP module.

For deprecated values, the MirrorLink Server shall not include them into the UPnP application listing. The MirrorLink Client shall ignore any content provided in deprecated elements.

Implementation Note:

MirrorLink 1.0 and 1.1 Servers may not include the *Signature* entry from *A_ARG_TYPE_AppList*. MirrorLink 1.0 and 1.1 Clients may ignore an existing *Signature* entry.

In case the advertised (child) application (e.g. "Musik") is part of a parent application (e.g. "RockScout"), the *variant* element shall define the unique *appID* of that parent application. The parent application shall be separately present in the UPnP application listing.

The parent's application identifier provided in the *variant* element, may then be announced instead of the originally launched child's *appID* within in the framebuffer context information, or through the UPnP application status. This means that a MirrorLink Client may see the parent application getting into the foreground on the MirrorLink Server, even when one of its child applications had been initially launched. The launched (child) application is linked to the announced (parent) foreground application via the *variant* field in the client's application list entry. A parent application shall not be the child of another application, i.e. nesting is not supported.

The *protocolID* element in A_ARG_TYPE_AppList is a string formatted as UTF-8 XML represents the remote access protocol of a specific application, which can be controlled remotely. Table 4-4 specifies the supported remote access protocols, supported from the TmApplicationServer:1 service.

Table 4-4: Supported Remote Access Protocols

protocolID	Protocol Name and Description
VNC	Virtual Networking Computing
RTP	Real Time Protocol
BTA2DP	Bluetooth Advanced Audio Distribution Profile
BTHFP	Bluetooth Hands Free Profile
DAP	Device Attestation Protocol
CDB	Common Data Bus
WFD (MirrorLink 1.2)	Wi-Fi Display
NONE	Used to indicate that application does not have any additional out-of-band connection using a remote access protocol
<VendorName-ProtocolName>	Vendor Specific Protocol Name. Note that the vendor name shall be appended in front of the protocol name and separated by an '-' (Hyphen)

The *format* element in *A_ARG_TYPE_AppList* is a string formatted as UTF-8 XML represents additional format information for dedicated remote access protocols. Table 4-5 specifies the *Remote Access Protocol Format* information.

Table 4-5: Remote Access Protocol Format

protocolID	Remote Access Protocol Format description
VNC	Not used
RTP	Comma separated list of supported RTP payload types. **Default**: "99"
BTA2DP	Not used
BTHFP	Not used
DAP	MirrorLink Version number **Allowed Values**: "1.0" or "1.1" or "1.2" or "1.3" **Default**: "1.3"
CDB	Version number of the CDB protocol **Allowed Values**: "1.1" **Default**: "1.1"
WFD (MirrorLink 1.2)	Not used
NONE	Not used
<VendorName-ProtocolName>	Vendor Specific

The *A_ARG_TYPE_AppList* contains many optional elements. Elements, which are used for specific remote access protocols, are given in Table 4-6.

Table 4-6: Used Elements in A_ARG_TYPE_AppList

Remote Access Protocol	VNC	DAP	CDB	BTHFP BTA2DP	RTP	WFD
remotingInfo	Used	Used	Used	Used	Used	Used
appInfo	Used	Used	Used	-	Used	Used
displayInfo	Used	-	-	-	-	Used
audioInfo	Used	-	-	Used	Used	Used
resourceStatus	Used	-	-	Used	-	Used
MirrorLink Version	≥ 1.0	≥ 1.0	≥ 1.1	≥ 1.1	≥ 1.1	≥ 1.2

An application with a *protocolID* of "VNC" may use the *resourceStatus* value "NA" to indicate that the particular application is accessible to launch, via a separately advertised Home Screen application.

4.2.8 A_ARG_TYPE_URI

A string encoded as UTF-8 represents a URI according to the following format, given in [2]:

```
   foo://example.com:8042/over/there?name=ferret#nose
   \_/   \______________/\_________/ \_________/ \__/
    |           |            |            |        |
 scheme     authority       path        query   fragment
```

with the authority being defined as

```
   example.com:8042
   \_________/ \__/
       |        |
     host     port
```

Implementation Note:

According to [2], the *scheme* and *host* values are case-incentive.

The values of *scheme*, *host* and *port* fields will differ based on the specific remoting protocol being used, as given in Table 4-7. The *port* field is optional for BTHFP and BTA2DP protocol identifiers. The *port* field shall be present for other non-vendor specific remote protocol identifiers. A_ARG_TYPE_URI is not specified for any vendor specific remote protocol identifier.

Table 4-7: URI Field values for Supported Remote Access Protocols

scheme	host	port	path, query, fragment
VNC	IP address of the VNC server (MANDATORY)	Port number of the VNC server (MANDATORY)	Not used
DAP	IP address of the DAP server (MANDATORY)	Port number of the DAP server (MANDATORY)	Not used
RTP	IP address of the RTP server or client (MANDATORY)	Port number of the RTP server or client (MANDATORY)	Not used
BTA2DP	ASCII string of the 48-bit Bluetooth address in hexadecimal notation (MANDATORY)	Stream End Point Identifier (SEID) in hexadecimal notation (OPTIONAL)	Not used
BTHFP	ASCII string of the 48-bit Bluetooth address in hexadecimal notation (MANDATORY)	RFCOMM channel in hexadecimal notation (OPTIONAL)	Not used
CDB	IP address of the CDB endpoint (MANDATORY)	Port number of the CDB endpoint (MANDATORY)	Not used
http	IP address of the resource being accessed	Port number of the resource being accessed	Used shall be available, in case the URI is used via HTTP-GET to retrieve the resource
WFD	IP address of the RTSP Server (MirrorLink 1.2) (MANDATORY)	Port number of the RTSP Server (MANDATORY)	Not used

The MirrorLink client shall use the *http* schema to the *host* and *port* of the UPnP Application Server Service's URL, if all *schema*, *host* and *port* entries are missing from the URI.

The URI shall be local to the MirrorLink Server. Access to external resources is not allowed.

4.2.9 A_ARG_TYPE_String

A simple string type (UTF-8).

4.2.10 A_ARG_TYPE_Bool

A simple Boolean string which can either have the value `'true'` or `'false'`.

4.2.11 A_ARG_TYPE_INT

A simple unsigned 32-bit integer represented in decimal (base 10) format.

4.2.12 A_ARG_TYPE_AppCertificateInfo

A string formatted as UTF-8 XML representing an application certificate. The format is given in Table 4-8. Its structure is given in Table 4-8.

Table 4-8: Structure of A_ARG_TYPE_AppCertificateInfo

Element	Description	Parent	Availability
certification	Application certification information	-	Optional
appID	Application identifier (appID) of the requested application MirrorLink Client shall check, whether the appID is equal to the one requested	certification	Required
nonce	Random DAP nonce, provided from the MirrorLink Client during the last DAP request 20-byte random number Base64-encoded Empty string indicates, that MirrorLink Client has not used DAP during the active MirrorLink session (A_ARG_TYPE_String)	certification	Required
appUUID	UUID of the application Unique application identifier (A_ARG_TYPE_String)	certification	Optional
entity*	Certifying entity (e.g. CCC, car OEM, HU OEM) The application shall be considered not certified, if this field is not available	certification	Optional
name	Entity name CCC certified applications shall have "CCC" as an entity name (A_ARG_TYPE_String)	entity	Required
targetList	Target **Default:** All targets are certified	entity	Optional
target*	Target name Comma separated list of MirrorLink Client vendor specific values. Might be interpreted as a White and/or a Black list (A_ARG_TYPE_String)	targetList	Required
restricted	Comma separated list of locales, where restricted use is certified (A_ARG_TYPE_String)	entity	Required
nonRestricted	Comma separated list of locales, where non-restricted use is certified (A_ARG_TYPE_String)	entity	Required
serviceList	List of allowed data services **Default:** All services are certified	entity	Optional
service+*	Service name Contains CDB service names, for which usage from the application has been certified (A_ARG_TYPE_String)	serviceList	Required
properties	Application properties Contains an UTF-8 XML representation of certified application properties. The XML representation is out of scope of the present document **Default:** Empty string (A_ARG_TYPE_String)	certification	Optional
Signature	XML signature over entire contents of the certification element. This is done as specified in [3] The key used in calculating the signature shall be the private part of the application-specific key which public part was bound to the attestation of UPnP-Server component. (The public part can be used to verify the signature.) The Reference element of the XML signature shall point to the certification element The SignatureMethod shall be RSA with SHA1. The KeyInfo element may be omitted. The mechanism for generation, exchange and maintainance of keys is out of scope for the present document	certification	Mandatory

The elements marked with a (*) can have multiple instances.

The elements marked with a (+) will have implementation specific values which may be outside the scope of the present document.

The appUUID shall be a universally-unique identifier for the application, across application versions and MirrorLink Server platforms. It shall begin with "uuid:" followed by a 128 bit number that shall be formatted as specified by the following grammar (taken from [1]):

```
UUID     = 4 * <hexOctet> "-" 2 * <hexOctet> "-"
           2 * <hexOctet> "-" 2 * <hexOctet> "-"
           6 * <hexOctet>
hexOctet = <hexDigit> <hexDigit>
hexDigit = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9"|
           "a"|"b"|"c"|"d"|"e"|"f"|"A"|"B"|"C"|"D"|"E"|"F"
```

The following is an example of an appUUID:

```
"uuid:2fac1234-31f8-11b4-a222-08002b34c003"
```

The entity name shall be used case-insensitive, when comparing the entry with other values, i.e. the following entries are identical:

1) "CCC" and "CcC"

2) "VW" and "vw"

3) "Volkswagen" and "VolksWagen"

A certificate may be valid for only a limited set of localities. The allowed localities, for each driving mode (i.e. *restricted* or *nonRestricted*) shall be listed in the dedicated section, separated by comma. Allowed localities are given below. The 3 letter abbreviations are taken from the IOC country codes.

- "EU" European Union member states
- "EPE" Europe (including Turkey) without countries listed separately or EU member states
- "RUS" Russia (may be included in "EPE" as well) - added in MirrorLink 1.3
- "CAN" Canada
- "USA" USA
- "BRA" Brazil (may be included in "AMERICA" as well) - added in MirrorLink 1.3
- "AMERICA" Americas without countries listed separately
- "AUS" Australia
- "KOR" Korea
- "JPN" Japan
- "CHN" China
- "HKG" Hongkong
- "TPE" Taiwan
- "IND" India
- "APAC" APAC states without countries listed separately
- "AFRICA" African countries without countries listed separately
- "WORLD" All countries

The list of locales may be extended in future versions of the specification. The localities shall be used case-insensitive, when comparing them with other values.

The list of locales, for which a certificate is valid, shall always include all localities, even if one locale includes other ones.

In order to show that an application has been drive-certified for the entire world, i.e. all localities, the `<restricted>` entry within A_ARG_TYPE_AppCertificateInfo shall have the following value:

```
"EU,EPE,BRA,CAN,USA,AMERICA,AUS,KOR,JPN,CHN,HKG,RUS,TPE,IND,APAC,
AFRICA,WORLD"
```

A value of `"WORLD"` is invalid.

4.3 Eventing and Moderation

Table 4-9 lists the eventing and moderation properties for each of the service state variables.

Table 4-9: Eventing and Moderation

Variable Name	Evented	Moderated Event	Max. Event Rate	Logical Relation	Min. Delta per Event
AppStatusUpdate	Yes	No	N/A	N/A	N/A
AppListUpdate	Yes	No	N/A	N/A	N/A
A_ARG_TYPE_AppStatus	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_AppID	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_ProfileID	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_URI	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_AppList	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_String	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_Bool	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_INT	No	N/A	N/A	N/A	N/A
A_ARG_TYPE_AppCertificateInfo	No	N/A	N/A	N/A	N/A

4.4 Supporting Multiple Client Profiles

Support for multiple Client Profiles is reserved for future use.

4.5 Actions

4.5.1 General

Table 4-10 lists the actions supported by the *TmApplicationServer* service.

Table 4-10: TmApplicationServer Service Actions

Name	Device R/O (see note 1)	Control Point R/O (see note 2)
GetApplicationList	R	R
LaunchApplication	R	R
GetApplicationStatus	R	O
TerminateApplication	R	O
GetApplicationCertificateInfo	R	CR
GetCertifiedApplicationsList	R	CR
GetAppCertificationStatus	R	O
SetAllowedApplicationsList	R	R

Name	Device R/O (see note 1)	Control Point R/O (see note 2)
NOTE 1: For a device this column indicates whether the action needs to be implemented or not, where R = REQUIRED, O = OPTIONAL, CR = CONDITIONALLY REQUIRED, CO = CONDITIONALLY OPTIONAL, X = Non-standard, add -D when deprecated (e.g. R-D, O-D).		
NOTE 2: For a control point this column indicates whether a control point needs to be capable of invoking this action, where R = REQUIRED, O = OPTIONAL, CR = CONDITIONALLY REQUIRED, CO = CONDITIONALLY OPTIONAL, X = Non-standard, add -D when deprecated (e.g. R-D, O-D). CR: Mandatory, if MirrorLink Client supports Drive Mode.		

Implementation Note:

The MirrorLink specification expects MirrorLink Servers to respond as fast as possible to any SOAP request and to not take use of the 30 s SOAP timeout; e.g. the Android specification uses a timeout of 3 s for application launch. Some MirrorLink Clients will show a notification or error message, in case of excessive response times, e.g. 10 s.

4.5.2 GetApplicationList

4.5.2.1 General

The *GetApplicationList* action provides a list of applications, which can be launched and terminated remotely. The list includes details such as application name, icons, remoting protocol used, application content category and trust level.

4.5.2.2 Arguments

Table 4-11: Arguments for GetApplicationList

Argument	Direction	relatedStateVariable
AppListingFilter	IN	A_ARG_TYPE_String
ProfileID	IN	A_ARG_TYPE_ProfileID
AppListing	OUT	A_ARG_TYPE_AppList

Parameters:

AppListingFilter (A_ARG_TYPE_String) - Application Listing Filter. This parameter is used by the UPnP Control Point to limit the AppListing value to those applications which meet the filter parameters. It consists of a comma-separated list of A_ARG_TYPE_AppList schema elements, attributes and their values (see clause 5.3 for examples). If the value of the *AppListingFilter* parameter is equal to `"*"` (default value), all elements and attributes (including optional ones, when present) and their values, are returned in *AppListing*. If the value of the *AppListingFilter* parameter is equal to `""` (empty string), then it is considered to be equivalent to having the value `"*"`.

ProfileID (A_ARG_TYPE_ProfileID) - *ProfileID* of client profile. Reserved for future. Shall be set to `"0"`.

Return Value:

AppListing (A_ARG_TYPE_AppList) - Returns a list of applications which are available for remote control and access. The applications listed in *AppListing* can be controlled using the *LaunchApplication*, *TerminateApplication* and *GetApplicationStatus* actions.

4.5.2.3 Effect on State

None.

4.5.2.4 Errors

Table 4-12: Error Codes for GetApplicationList

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	Operation Rejected	The *TmApplicationServer* service has rejected the operation.
815	Device Locked	The action cannot be processed as the device hosting the *TmApplicationServer* service is locked. User needs to unlock the device first.
820	Invalid Argument	The argument passed is invalid.
830	Invalid Profile ID	The profile identifier does not exist or the application cannot use the specified profile identifier.

4.5.3 LaunchApplication

4.5.3.1 General

The *LaunchApplication* action enables a client to remotely start a specific application on the MirrorLink Server device and provide remote access to it if available. Note that this action can be used to launch all types of applications including daemons and other servers which need not have a UI.

In case the application is a UI application already running, invoking the *LaunchApplication* action again causes the application to be brought to the foreground and given control of the UI. Hence, this action can be used by the MirrorLink Client to switch between different remote applications.

The MirrorLink UPnP Server shall ensure that the implementation of the *LaunchApplication* action is idempotent. For example, if an application with a specific *AppID* is already running, then multiple calls to *LaunchApplication* using the same *AppID* will not launch the application again but it will put the application in the foreground. If the application being launched is a UI application, then the MirrorLink Server device shall give control of the UI to the launched application before returning a response to the *LaunchApplication* action.

The MirrorLink UPnP Control Point may launch a VNC Server directly, if advertised by the MirrorLink UPnP Server. In that case, MirrorLink Client should receive the MirrorLink Server's current screen content. The MirrorLink Client may use *LaunchApplication* to bring a particular application into the foreground. The MirrorLink UPnP Server may launch a stand-alone VNC Server together with the launch of another VNC based application. In this case, the MirrorLink UPnP Server shall notify the VNC Server's application status as either `foreground` or `background`.

The MirrorLink UPnP Control Point may launch a BT HFP or BT A2DP component directly, if advertised by the MirrorLink UPnP Server; a timeout may happen for *LaunchApplication* of those components, as user input may be needed for pairing and then connecting through BT HFP or BT A2DP. In that case, the MirrorLink Client may execute the *LaunchApplication* action again.

4.5.3.2 Arguments

Table 4-13: Arguments for LaunchApplication

Argument	Direction	relatedStateVariable
AppID	IN	A_ARG_TYPE_AppID
ProfileID	IN	A_ARG_TYPE_ProfileID
AppURI	OUT	A_ARG_TYPE_URI

Parameters:

AppID (A_ARG_TYPE_AppID) - Unique application ID of application to be launched by invocation of this action.

ProfileID (A_ARG_TYPE_ProfileID) - *ProfileID* of client profile. Reserved for future. Shall be set to "0".

Return Value:

AppURI (A_ARG_TYPE_URI) - This method will return a URI for accessing the remote application using the protocol identifier as given in the <remotingInfo> element of the *AppListing* returned in response to the *GetApplicationList* action.

4.5.3.3 Effect on State

This action affects the value of the *AppStatusUpdate* state variable if it contains an entry corresponding to the value of the *AppID* argument.

4.5.3.4 Errors

Table 4-14: Error Codes for LaunchApplication

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
810	Bad AppId	The AppId does not exist or is malformed.
811	Unauthorized AppId	The application identified by this AppId cannot be controlled or accessed remotely.
813	Launch Failed	Failed to launch the application.
814	Resource Busy	The requested application resource is busy, This error can occur when the resource is already busy and *resourceStatus* in the *AppListing* is set equal to "NA".
815	Device Locked	The action cannot be processed as the device hosting the *TmApplicationServer* service is locked. User needs to unlock the device first.
830	Invalid Profile ID	The profile identifier does not exist or the application cannot use the specified profile identifier.

4.5.4 TerminateApplication

4.5.4.1 General

The TerminateApplication action enables the client to remotely terminate any application, which is listed in the *AppList* returned by the *GetApplicationList* action. It shall be noted that any application listed as in the AppList can be terminated using this action even if it was not started through invocation of the *LaunchApplication* action. Furthermore, this TerminateApplication will have no effect on any applications which are not listed in AppList. The MirrorLink UPnP Control Point should terminate a BT HFP or BT A2DP component only, if the component has been previously launched using *LaunchApplication* or if the MirrorLink Client provided its Bluetooth MAC address (*bdAddr*) in A_ARG_TYPE_ClientProfile.

A MirrorLink Server need not support terminating an application (i.e. the application is removed from the MirrorLink Server device's process list), but the MirrorLink Server shall ensure the following behavior in case the MirrorLink has requested the application's termination:

- In case the application to be terminated, is a remote user interface application in the `"Foreground"`, that application shall change its application status to `"Notrunning"` or `"Background"`. The terminated application shall end any ongoing audio streaming. The MirrorLink Server may (potentially launch and) bring another application into the foreground.
- In case the application to be terminated, is a remote user interface application in the `"Background"`, that application shall change its application status to `"Notrunning"` or stay in `"Background"`. The terminated application shall end any ongoing audio streaming. The MirrorLink Server shall keep the current foreground applications.
- In case the application to be terminated, is a RTP Server, RTP Client, DAP endpoint or CDB Endpoint application, that application shall change its application status to `"Notrunning"`. The MirrorLink Server shall keep the current foreground applications.

If any of the above conditions is fulfilled, the MirrorLink Server shall respond with `"TerminationResult=true"`, otherwise with `"TerminationResult=false"`. The MirrorLink Server shall always notify any application's status change to the MirrorLink Client via the *AppStatusUpdate* evented variable, if that application has been included within the UPnP advertisements.

The MirrorLink UPnP Server shall ensure that the implementation of the *TerminateApplication* action is idempotent. For example, if an application with a specific *AppID* is not running, then further calls to *TerminateApplication* using the same *AppID* will have no effect but will return *TerminationResult* as `True`.

The client should not call the *TerminateApplication* action every time it switches to a different remote application. If a client launches a different remote application without terminating the existing one, the MirrorLink Server shall send the original application to the background.

4.5.4.2 Arguments

Table 4-15: Arguments for TerminateApplication

Argument	Direction	relatedStateVariable
AppID	IN	A_ARG_TYPE_AppID
ProfileID	IN	A_ARG_TYPE_ProfileID
TerminationResult	OUT	A_ARG_TYPE_Bool

Parameters:

AppID (A_ARG_TYPE_AppID) - Unique application ID of application to be terminated by invocation of this action.

ProfileID (A_ARG_TYPE_ProfileID) - *ProfileID* of the client profile. Reserved for future. shall be set to `"0"`.

Return Value:

TerminationResult (A_ARG_TYPE_Bool) - Returns true if application terminated successfully, false otherwise.

4.5.4.3 Effect on State

This action affects the value of the *AppStatusUpdate* state variable if it contains an entry corresponding to the value of the *AppID* argument.

4.5.4.4 Errors

Table 4-16: Error Codes for TerminateApplication

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
810	Bad AppId	The *AppId* does not exist or is malformed.
811	Unauthorized AppId	The application identified by this *AppId* cannot be controlled or accessed remotely.
815	Device Locked	The action cannot be processed as the device hosting the *TmApplicationServer* service is locked. User needs to unlock the device first.
830	Invalid Profile ID	The profile identifier does not exist or the application cannot use the specified profile identifier.

4.5.5 GetApplicationStatus

4.5.5.1 General

The *GetApplicationStatus* action provides the current status of one or all applications and allows the client to request automatic updates of status changes for any application listed in the *AppList* returned by the *GetApplicationList* action.

4.5.5.2 Arguments

Table 4-17: Arguments for GetApplicationStatus

Argument	Direction	relatedStateVariable
AppID	IN	A_ARG_TYPE_AppID
AppStatus	OUT	A_ARG_TYPE_AppStatus

Parameters:

AppID (A_ARG_TYPE_AppID) - Unique application ID; can be set equal to the string "*" to indicate a wildcard. In case the *AppID* parameter is set by the client to be equal to "*" then the MirrorLink UPnP Server shall return the application status of all applications, which can be controlled using the *TmApplicationServer* service.

Return Value

AppStatus (A_ARG_TYPE_AppStatus) - Status of application(s).

4.5.5.3 Effect on State

None.

4.5.5.4 Errors

Table 4-18: Error Codes for GetApplicationStatus

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
810	Bad AppId	The *AppId* does not exist or is malformed.
812	Cannot Determine Status	The status of the specified application(s) cannot be determined.
815	Device Locked	The action cannot be processed as the device hosting the *TmApplicationServer* service is locked. User needs to unlock the device first.

4.5.6 GetApplicationCertificateInfo

4.5.6.1 General

The *GetApplicationCertificateInfo* action provides certification data for one application. Certification data is provided, if the application certificate is successfully validated by the MirrorLink Server and has any certifying entity, independent of whether the certifying entities include a CCC- and/or any Member-certifying entity section.

The MirrorLink Clients will use the received information to decide, whether the application is certified under its conditions (e.g. locale, member certification, drive/base certification). The MirrorLink Client may use the *GetCertifiedApplicationsList* action and the *AppCertFilter* settings to pre-filter applications, as described in clause 5.4.

4.5.6.2 Arguments

Table 4-19: Arguments for GetApplicationCertificateInfo

Argument	Direction	relatedStateVariable
AppID	IN	A_ARG_TYPE_AppID
AppCertification	OUT	A_ARG_TYPE_AppCertificateInfo

Parameters:

AppID (A_ARG_TYPE_AppID) - Unique application ID for the application for which the certificate information should be returned.

Return Value

AppCertification (A_ARG_TYPE_AppCertificateInfo) - Certificate of application. An empty string is returned, if the application identified by the AppID does not have a valid certificate.

4.5.6.3 Effect on State

None.

4.5.6.4 Errors

Table 4-20: Error Codes for GetApplicationCertificateInfo

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
810	Bad AppId	The AppId does not exist or is malformed.
815	Device Locked	The action cannot be processed as the device hosting the TmApplicationServer service is locked. User needs to unlock the device first.

4.5.7 GetCertifiedApplicationsList

4.5.7.1 General

Get a list of certified applications, matching a set of criteria.

4.5.7.2 Arguments

Table 4-21: Arguments for GetCertifiedApplicationsList

Argument	Direction	relatedStateVariable
AppCertFilter	IN	A_ARG_TYPE_String
ProfileID	IN	A_ARG_TYPE_ProfileID
CertifiedAppList	OUT	A_ARG_TYPE_String

Parameters:

AppCertFilter (A_ARG_TYPE_String) - Application Certification Filter. This parameter is used by the UPnP Control Point to limit the CertifiedAppList to those applications which meet the filter parameters. It consists of a comma-separated list of A_ARG_TYPE_AppCertificateInfo schema elements, attributes and their values (see clause 5.4 for examples). If the value of the AppListingFilter parameter is equal to `""` (empty string), then it is considered to be equivalent to having the value `"*"(default value)`.

ProfileID (A_ARG_TYPE_ProfileID) - ProfileID of client profile. Reserved for future. shall be set to `"0"`.

Return Value

CertifiedAppList (A_ARG_TYPE_String) - Comma separated list of application identifiers of type A_ARG_TYPE_AppId, which are certified based on the giving input filter values. An empty string is returned, if no application matches the input filter criteria.

4.5.7.3 Effect on State

None.

4.5.7.4 Errors

Table 4-22: Error Codes for GetCertifiedApplicationsList

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
815	Device Locked	The action cannot be processed as the device hosting the TmApplicationServer service is locked. User needs to unlock the device first.
820	Invalid Argument	The argument passed is invalid.
830	Invalid Profile ID	The profile identifier does not exist or the application cannot use the specified profile identifier.

4.5.8 GetAppCertificationStatus

4.5.8.1 General

Return the certification status of a given application, under the provided properties.

4.5.8.2 Arguments

Table 4-23: Arguments for GetAppCertificationStatus

Argument	Direction	relatedStateVariable
AppID	IN	A_ARG_TYPE_AppID
AppCertFilter	IN	A_ARG_TYPE_String
ProfileID	IN	A_ARG_TYPE_ProfileID
AppCertified	OUT	A_ARG_TYPE_Bool

Parameters:

AppID (A_ARG_TYPE_AppID) - Unique application ID for the application, for which the certication status should be returned.

AppCertFilter (A_ARG_TYPE_String) - Application Listing Filter. This parameter is used by the UPnP Control Point to list the criteria to check the application against. It consists of a comma-separated list of A_ARG_TYPE_AppCertificateInfo schema elements, attributes and their values (see clause 5.4 for examples). If the value of the AppListingFilter parameter is equal to `"*"` (default value), the application passes this parameter. If the value of the AppListingFilter parameter is equal to `""` (empty string), then it is considered to be equivalent to having the value `"*"`.

ProfileID (A_ARG_TYPE_ProfileID) - ProfileID of client profile. Reserved for future. shall be set to `"0"`.

Return Value

AppCertified (A_ARG_TYPE_Bool) - Boolean value, whether the application is certified (`"true"`) or not (`"false"`);

4.5.8.3 Effect on State

None.

4.5.8.4 Errors

Table 4-24: Error Codes for GetAppCertificationStatus

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
810	Bad AppId	The AppId does not exist or is malformed.
815	Device Locked	The action cannot be processed as the device hosting the TmApplicationServer service is locked. User needs to unlock the device first.
820	Invalid Argument	The argument passed is invalid.

4.5.9 SetAllowedApplicationsList

4.5.9.1 General

The *SetAllowedApplicationsList* action allows the MirrorLink Client to inform the MirrorLink Server about the list of supported UI applications in drive and park mode. The MirrorLink Client shall invoke the *SetAllowedApplicationsList* action after it has retrieved the initial UPnP application list and analyzed the application's certification status, before launching any UI application.

The MirrorLink Client shall invoke the *SetAllowedApplicationsList* action, whenever the allowed application list changes, due to an *AppListUpdate* event, if the list of allowed applications has changed.

The MirrorLink Client shall not use the *SetAllowedApplicationsList* action with MirrorLink 1.0, 1.1 and 1.2 Servers. MirrorLink 1.3 Servers being connected to a MirrorLink 1.0, 1.1 or 1.2 Client should use the MirrorLink Client's usage of the *AppCertFilter* to understand allowed applications.

4.5.9.2 Arguments

Table 4-25: Arguments for SetAllowedApplicationsList

Argument	Direction	relatedStateVariable
AllowedAppListNonRestricted	IN	A_ARG_TYPE_String
AllowedAppListRestricted	IN	A_ARG_TYPE_String
ProfileID	IN	A_ARG_TYPE_ProfileID

Parameters:

AllowedAppListNonRestricted (*A_ARG_TYPE_String*) - Comma separated list of application identifiers of type *A_ARG_TYPE_AppId*, which are allowed for non-restricted driving mode. If the value of the *AllowedAppListNonRestricted* parameter is equal to "*" (default value), all applications from the provided application list are allowed. An *AllowedAppListNonRestricted* parameter value, equal to "" (empty string), defines an empty list, in which case the MirrorLink UPnP Control Point does not allow any application.

AllowedAppListRestricted (*A_ARG_TYPE_String*) - Comma separated list of application identifiers of type *A_ARG_TYPE_AppId*, which are allowed for restricted driving mode. If the value of the *AllowedAppListRestricted* parameter is equal to `"*"` (default value), all applications from the provided application list are allowed. An *AllowedAppListRestricted* parameter value, equal to `""` (empty string), defines an empty list, in which case the MirrorLink UPnP Control Point does not allow any application.

ProfileID (*A_ARG_TYPE_ProfileID*) - ProfileID of client profile. Reserved for future. shall be set to `"0"`.

Return Value

None.

4.5.9.3 Effect on State

None.

4.5.9.4 Errors

Table 4-26: Error Codes for SetAllowedApplicationsList

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
810	Bad AppID	One AppID does not exist or is malformed.
815	Device Locked	The action cannot be processed as the device hosting the TmApplicationServer service is locked. User needs to unlock the device first.
820	Invalid Argument	The argument passed is invalid.
830	Invalid Profile ID	The profile identifier does not exist or the application cannot use the specified profile identifier.

4.5.10 Relationships Between Actions

Figure 4-1 shows the relationship between service actions invoked using the same client profile. Note that the *GetApplicationList* action has no relationship with the other actions. Also note that actions invoked using different client profiles will have no relationship to each other.

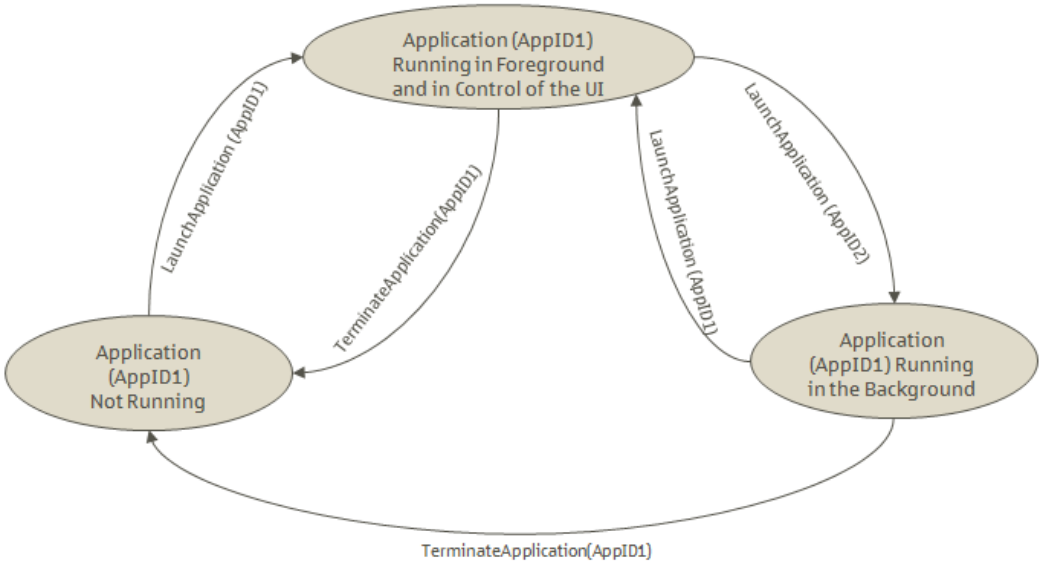

Figure 4-1: Relationship between actions invoked using the same client profile

4.5.11 Error Code Summary

Table 4-27 lists error codes common to actions for this service type. If an action results in multiple errors, the most specific error should be returned.

Table 4-27: Error Code Summary

ErrorCode	errorDescription	Description
400-499	TBD	See UPnP Device Architecture section on Control.
500-599	TBD	See UPnP Device Architecture section on Control.
600-699	TBD	See UPnP Device Architecture section on Control.
701	Operation Rejected	The TmApplicationServer service has rejected the operation. MirrorLink Client should retry the action.
810	Bad AppId	The AppId does not exist or is malformed. MirrorLink Client should check the appId (e.g. using GetApplicationList) and retry action. MirrorLink Client should not retry the action with the same appId.
811	Unauthorized AppId	The application identified by this AppId cannot be controlled or accessed remotely. MirrorLink Client should not retry the action.
812	Cannot Determine Status	The status of the specified application(s) cannot be determined. MirrorLink Client should retry the action.
813	Launch Failed	Failed to launch the application. MirrorLink Client should retry the action.
814	Resource Busy	The requested application resource is busy, This error can occur when the resource is already busy and resourceStatus in the AppListing is set equal to "NA". MirrorLink Client should retry the action. MirrorLink Client should retry the action only after receiving notification that the resource is becoming free.
815	Device Locked	The action cannot be processed as the device hosting the TmApplicationServer service is locked. User needs to unlock the device first. MirrorLink Client should not retry the action.

ErrorCode	errorDescription	Description
820	Invalid Argument	The argument passed is invalid. The MirrorLink Client should verify the format of the arguments. MirrorLink Client should not retry the action with the same arguments.
830	Invalid Profile ID	The profile identifier does not exist or the application cannot use the specified profile identifier. MirrorLink Client should check the client profile (GetClientProfile) and its application support from the GetApplicationList response, and retry the action. MirrorLink Client should not retry the action with the same arguments.

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

The MirrorLink Client should give up after 3 retry attempts. Notification about (finally) failing a UPnP action may be necessary. The specification of notification requirements is outside the scope of the present document.

5 Theory of Operation

5.1 Use of Quotation Marks

Throughout the present document, two kinds of quotation marks are used:

- Quotation marks as in `“words”` are used to highlight a textual element, for readability purpose only. The quotation marks shall not be used within XML schemata, or within arguments of SOAP actions.
- Quotation marks as in `"music"` are part of the XML or SOAP syntax and shall be maintained.

Example:

If <protocolId> shall be `“VNC”`, then:

- `<protocolId>VNC</protocolId>` is valid XML and
- `<protocolId>“VNC”</protocolId>` is invalid XML.

NOTE: A MirrorLink 1.0 compliant Client may incorrectly include the `“”` quotation marks within an *AppListingFilter* parameter. A MirrorLink Server should ignore those.

5.2 Identification of Applications from A_ARG_TYPE_AppList

5.2.1 Identifying the VNC Server

If the MirrorLink Server supports a stand-alone VNC Server it shall set the following entries, so that it can be identified within the A_ARG_TYPE_AppList response to a *GetApplicationList* action from the MirrorLink Client:

- <protocolId> shall be `“VNC”.`
- <appCategory> shall be `“0xF0000001”` (Server functionality).

The MirrorLink server shall not advertise more than one stand-alone VNC server service for each framebuffer. If the MirrorLink server has multiple framebuffers, the corresponding VNC servers shall be advertised in dimension order, i.e. framebuffers with bigger dimension appear first.

5.2.2 Identifying Remote VNC based Applications

The MirrorLink Server shall set the following entries for all VNC based remote applications, so that they can be identified within the A_ARG_TYPE_AppList response to a *GetApplicationList* action from the MirrorLink Client:

- <protocolId> shall be "VNC".

5.2.3 Identifying Audio Links

The MirrorLink Server shall set the following entries for remote components, providing an audio link, so that they can be identified within the *A_ARG_TYPE_AppList* response to a *GetApplicationList* action from the MirrorLink Client.

The following elements within *A_ARG_TYPE_AppList* shall be provided.

Table 5-1

	<protocolID>	<appCategory>	<audioType>	<direction>
RTP Server	"RTP"	"0xF0000001"	"phone", "application" or "all"	"out"
RTP Client		"0xF0000002"		"in"
BT HFP	"BTHFP"	-	"phone"	"bi"
BT A2DP	"BTA2DP"		"application"	"out"
NOTE: Phone audio includes Voice and Command Control audio, besides cellular call audio.				

All other optional elements may be omitted.

NOTE 1: MirrorLink 1.0 Server device may omit the audioType entry for BT HFP, BT A2DP and RTP Clients and Servers. Therefore BT HFP will be considered as <audioType> being equal to "phone" and BT A2DP as <audioType> being equal to "application". For RTP Clients and Servers the <audioType> will be considered being equal to "application".

NOTE 2: MirrorLink 1.0 Server device may omit the direction entry as well as the *appCategory* entry. To safely distinguish between an RTP Server and Client, consider the direction's entry default value "out". The MirrorLink server will not advertise multiple RTP servers or clients with different RTP payload types. The MirrorLink server will not advertise more than one BT HFP and BT A2DP component. The MirrorLink Server may advertise RTP Server or Clients, supporting RTP payload types not being supported from the MirrorLink Client. The MirrorLink Client will launch an RTP server or RTP client, which is advertising support for at least one RTP payload type supported from the MirrorLink Client's RTP counterpart.

NOTE 3: Some older MirrorLink Client will pick the first advertised RTP endpoint as the RTP Server, even if the endpoint is actually an RTP Client. Therefore, a MirrorLink Server should list the RTP Server with RTP payload type 99 before RTP Servers with other payload types and before RTP Clients within the UPnP *A_ARG_TYPE_AppList*. Otherwise, a forward audio connection may not be established with these older MirrorLink Clients.

The A_ARG_TYPE_AppList audioInfo entry provides information, which audio related services are available from the MirrorLink Server:

- **Phone Call (over RTP):** The advertised RTP Client **and** RTP Server shall have an audioType of "phone" or "all" **and** shall include the "Phone Audio" flag within the audioInfo@contentCategory.
- **Phone Call (over BT):** The advertised BT HFP component shall have an audioType of "phone" **and** shall include the "Phone Audio" flag within the audioInfo@contentCategory.
- **Voice Command (over RTP):** The advertised RTP Client shall have an audioType of "phone" or "all" **and** shall include the "Voice Command In" flag within the audioInfo@contentCategory.
- **Voice Command (over BT):** The advertised BT HFP component shall have an audioType of "phone" **and** shall include the "Voice Command In" flag within the audioInfo@contentCategory.

- **Media Streaming (over RTP):** The advertised RTP Server shall have an `audioType` of `"application"` or `"all"` **and** shall include the `"Media Audio Out"` flag within the `audioInfo@contentCategory`.
- **Media Streaming (over BT):** The advertised BT A2DP component shall have an `audioType` of `"application"` **and** shall include the `"Media Audio Out"` flag within the `audioInfo@contentCategory`.

In case a Bluetooth component's resource status is marked as `"busy"` or `"NA"` in the A_ARG_TYPE_AppList response, the MirrorLink Client should subscribe to the AppListUpdate status variable to receive the information, when the resource becomes `"free"`.

5.2.4 Identifying Common Data Bus

If the MirrorLink Server supports the Common Data Bus, it shall include the Common Data Bus endpoint as an application within A_ARG_TYPE_AppList. In that list, it shall set the following entries, so that the Common Data Bus endpoint can be identified from the MirrorLink Client.

The MirrorLink server shall set the following entries to the designated values:

- <protocolId> shall be `"CDB"`.
- <appCategory> shall be `"0xF0000000"` or non-existing.

All other optional elements may be omitted.

The MirrorLink server shall not advertise more than one Common Data Bus endpoint with the same major version.

5.2.5 Identifying Device Attestation Protocol Server

If the MirrorLink Server supports the Device Attestation Protocol, it shall include the Device Attestation Protocol Server as an application within A_ARG_TYPE_AppList. In that list, it shall set the following entries, so that the Device Attestation Protocol Server can be identified from the MirrorLink Client:

- <protocolId> shall be `"DAP"`.
- <appCategory> shall be `"0xF0000001"` (Server functionality).

All other optional elements may be omitted.

The MirrorLink server shall not advertise more than one Device Attestation Protocol server with the same version number.

5.3 Example Values of AppListingFilter

The GetApplicationList action returns application listings that are a match for the parameters specified in *AppListingFilter*.

AppListingFilter is composed of a comma-separated list of *A_ARG_TYPE_AppList* schema elements, attributes and their values. For clarification, the commas in the filter expression are equivalent to an AND operation. For example, `"name="music",description="*""` is interpreted as the condition `"name="music""` AND `"description="*""`. Optional as well as REQUIRED schema elements can be included in the *AppListingFilter*.

To identify an element in the XML hierarchy, the filter should use the shortest path in the hierarchy, which uniquely identifies the element, according to the following Extended Backus-Naur Form (EBNF):

```
element = [ { parent-element , "@" } ] , child-element;
```

The symbol `"@"` hereby acts as the delimiter between parent and child elements.

Valid examples are given below:

- `"*"` (no filtering applied)
- `"protocolId="VNC""` (filter for VNC applications)
- `"protocolId="RTP""` (filter for RTP endpoints)
- `"protocolId="BTA2DP""` (filter for Bluetooth A2DP endpoints)
- `"protocolId="BTHFP""` (filter for Bluetooth HFP endpoints)
- `"protocolId="DAP""` (filter for DAP endpoints)
- `"protocolId="CDB""` (filter for CDB endpoints)
- `"protocolId="WFD""` (filter for WFD applications)
- `"protocolId="VNC",appCategory="0xF0000001""` (filter for VNC Server endpoint)
- `"protocolId="RTP",appCategory="0xF0000001""` (filter for RTP Server endpoint)
- `"protocolId="RTP",appCategory="0xF0000002""` (filter for RTP Client endpoint)
- `"protocolId="DAP",appCategory="0xF0000001""` (filter for DAP Server endpoint)
- `"protocolId="CDB",appCategory="0xF0000000""` (filter for CDB Endpoint)
- `"appID="`*`appID_value`*`""` (filter for specific application with respective application identifier; *appID_value* shall be replaced with the application identifier value as provided in previous UPnP app listings or UPnP GENA events.)

The MirrorLink Server shall support above *AppListingFilter* examples, where *protocolId* may also have other allowed values. The MirrorLink Server may ignore parts of more complex *AppListingFilter* values by removing non-supported filter elements or combinations of them.

E.g. a MirrorLink Server may simplify the *AppListingFilter*:

`"protocolId="VNC",name="music""`

to

`"protocolId="VNC"".`

The values of schema elements and attributes specified in *AppListingFilter* are not considered to be case-sensitive for purposes of matching. For example, an AppListingFilter equal to `"name="music""` will return listings of applications which have `"Music"`, `"mUSic"` or `"music"` in their names.

MirrorLink 1.3 Clients shall not use spaces or other white spaces within the *AppListingFilter*, unless spaces are added purposely to the XML element's value between the quotation marks, like

`"name="This OEM",nonRestricted="EU""`

For backward compatibility, MirrorLink 1.3 Servers shall ignore (i.e. trim) any of these unintended white spaces.

5.4 Example Values of AppCertFilter

The *GetCertifiedApplicationsList* action returns a list of application that are certified with respect to the parameter specified in the *AppCertFilter*.

AppCertFilter is composed of a comma-separated list of *A_ARG_TYPE_AppCertificateInfo* schema elements, attributes and their values. For clarification, the commas in the filter expression are equivalent to an AND operation. In case all comma-separated elements within the filter expression belong to the same parent, then all given conditions shall apply for the same parent instance, in case multiple instances exist. The MirrorLink Server shall apply the *AppCertFilter* only to the *entity* elements.

Optional as well as REQUIRED schema elements can be included in the *AppCertFilter*. To identify an element in the XML hierarchy, the filter should use the shortest path in the hierarchy, which uniquely identifies the element, according to the following Extended Backus-Naur Form (EBNF):

```
element = [ { parent-element , “@” } ] , child-element;
```

The symbol "@" hereby acts as the delimiter between parent and child elements.

Valid examples are given below:

- An *AppCertFilter* set to `“*”` returns all applications, which are base or drive certified for at least one locale.
- An *AppCertFilter* set to `“restricted="EU"”` returns all applications, which are drive-certified for EU.
- An *AppCertFilter* set to `“name="CCC"”` returns all applications, which are CCC base or drive certified.
- An *AppCertFilter* set to `“name="OEM-A"”` returns all applications, which are member base or drive certified by OEM-A.
- An *AppCertFilter* set to `“name="CCC",restricted="EU"”` returns all applications, which are CCC drive-certified for EU.
- An *AppCertFilter* set to `“name="CCC",restricted="USA"”` returns all applications, which are CCC drive-certified for USA.
- An *AppCertFilter* set to `“name="CCC",restricted="JPN"”` returns all applications, which are CCC drive-certified for Japan.
- An *AppCertFilter* set to `“name="CCC",nonRestricted="EU"”` returns all applications, which are CCC base-certified for EU.
- An *AppCertFilter* set to `“name="OEM-A",restricted="EU",target="`*xyz*`"”` returns all applications, which are drive-certified for EU by OEM-A CCC base-certified for EU, containing the target *xyz*. Use of *targetList* is only allowed for member-certified applications.

The MirrorLink Server shall support above *AppCertFilter* examples, where *name*, *restricted* and *nonRestricted* may also have other allowed values. The MirrorLink Server may ignore parts of more complex *AppCertFilter* values by removing non-supported filter elements or combinations of them.

The default value of *AppCertFilter* shall be set equal to `“*”`.

The values of schema elements and attributes specified in *AppCertFilter* are not considered to be case-sensitive for purposes of matching.

MirrorLink 1.3 Clients shall not use spaces or other white spaces within the *AppCertFilter*, unless spaces are added purposely to the XML element's value between the quotation marks, like

```
“name="This OEM",restricted="EU",target="My Target"”
```

For backward compatibility, MirrorLink 1.3 Servers shall ignore (i.e. trim) any of these unintended white spaces.

5.5 Example Values of State Variables

5.5.1 AppStatusUpdate

The AppStatusUpdate state variable consists of a comma-separated list of appIDs identifying applications whose status has changed. Each entry in the list is of the type A_ARG_TYPE_AppID.

AppStatusUpdate value will consist of a comma separated list of all application identifiers (appIDs) of applications listed in A_ARG_TYPE_AppList when the event is issued by the TmApplicationServer service for the first time.

EXAMPLE: "0xef128320, 0xffff8320,0x12f128320,0x00128320".

5.5.2 AppListUpdate

The AppListUpdate state variable consists of a comma-separated list of appIDs identifying applications whose entries in the application list have changed. Each entry in the list is of the type A_ARG_TYPE_AppID.

AppListUpdate value will consist of a comma separated list of all application identifiers (appIDs) of applications listed in A_ARG_TYPE_AppList when the event is issued by the TmApplicationServer service for the first time.

EXAMPLE: "0xffff8320,0xef128320, 0x00128320".

5.5.3 A_ARG_TYPE_AppStatus

The value of the non-evented state variable *A_ARG_TYPE_AppStatus* is an XML block corresponding to a list of applications whose status updates have been requested by the control point. The following examples illustrate some possible values of A_ARG_TYPE_AppStatus.

EXAMPLE 1: If the application status of a single application with AppID "App_1" is returned.

```
<?xml version="1.0" encoding="UTF-8"?>
<appStatusList>
    <appStatus>
        <appID>0x01</appID>
        <status>
            <profileID>2</profileID>
            <statusType>Foreground</statusType>
        </status>
    </appStatus>
</appStatusList>
```

EXAMPLE 2: If the application status of multiple applications is returned, for example when AppID is set equal to "*" during invocation of the GetApplicationStatus action.

```
<?xml version="1.0" encoding="UTF-8"?>
<appStatusList>
    <appStatus>
        <appID>0x01</appID>
        <status>
            <profileID>2</profileID>
            <statusType>Foreground</statusType>
        </status>
    </appStatus>
    <appStatus>
        <appID>0x02</appID>
        <status>
            <profileID>2</profileID>
            <statusType>Background</statusType>
        </status>
        <status>
            <profileID>3</profileID>
            <statusType>Foreground</statusType>
        </status>
    </appStatus>
    <appStatus>
        <appID>0x03</appID>
        <status>
            <profileID>2</profileID>
            <statusType>Notrunning</statusType>
```

```
        </status>
    </appStatus>
</appStatusList>
```

5.5.4 A_ARG_TYPE_AppList

The value of the non-evented state variable *A_ARG_TYPE_AppList* is an XML block which lists all the applications which can be remotely controlled by this service. It is returned in response to a *GetApplicationList* action.

```
<?xml version="1.0" encoding="UTF-8"?>
<appList xml:id="mlServerAppList">
    <app>
        <appID>0x5678</appID>
        <name>Navigation</name>
        <providerName>Nokia</providerName>
        <providerURL>http://www.nokia.com/maps</providerURL>
        <description>Mobile Navigation Application</description>
        <iconList>
            <icon>
                <mimetype>image/png</mimetype>
                <width>40</width>
                <height>60</height>
                <depth>24</depth>
                <url>/icons/icon5678.png</url>
            </icon>
        </iconList>
        <remotingInfo>
            <protocolID>VNC</protocolID>
        </remotingInfo>
        <appCertificateURL>
            http://192.168.100.1/navApp.cert
        </appCertificateURL>
        <appInfo>
            <appCategory>0x00050000</appCategory>
            <trustLevel>0x0080</trustLevel>
        </appInfo>
        <displayInfo>
            <contentCategory>0x00010028</contentCategory>
            <trustLevel>0x0080</trustLevel>
        </displayInfo>
        <audioInfo>
            <audioType>application</audioType>
            <contentCategory>0x02</contentCategory>
            <trustLevel>0x0080</trustLevel>
        </audioInfo>
    </app>
    <app>
        <appID>0x01</appID>
        <name>RockScout</name>
        <description>Music Player</description>
        <iconList>
            <icon>
                <mimetype>image/png</mimetype>
                <width>128</width>
                <height>128</height>
                <depth>24</depth>
                <url>/icons/icon01.png</url>
            </icon>
        </iconList>
        <remotingInfo>
            <protocolID>VNC</protocolID>
        </remotingInfo>
        <appCertificateURL>
            http://192.168.100.1/app01.cert
        </appCertificateURL>
        <appInfo>
            <appCategory>0x00030001</appCategory>
            <trustLevel>0x0080</trustLevel>
        </appInfo>
        <displayInfo>
            <contentCategory>0x00010020</contentCategory>
            <trustLevel>0x0080</trustLevel>
        </displayInfo>
        <audioInfo>
            <audioType>application</audioType>
            <contentCategory>0x00000002</contentCategory>
```

```
        <trustLevel>0x0080</trustLevel>
    </audioInfo>
</app>
<app>
    <appID>0x02</appID>
    <name>Spotify</name>
    <variant>0x01</variant>
    <description>Spotify for RockScout</description>
    <iconList>
        <icon>
            <mimetype>image/png</mimetype>
            <width>128</width>
            <height>128</height>
            <depth>24</depth>
            <url>/icons/icon02.png</url>
        </icon>
    </iconList>
    <remotingInfo>
        <protocolID>VNC</protocolID>
    </remotingInfo>
    <appCertificateURL>
        http://192.168.100.1/app02.cert
    </appCertificateURL>
    <appInfo>
        <appCategory>0x00030002</appCategory>
        <trustLevel>0x0080</trustLevel>
    </appInfo>
    <displayInfo>
        <contentCategory>0x00010020</contentCategory>
        <trustLevel>0x0080</trustLevel>
    </displayInfo>
    <audioInfo>
        <audioType>application</audioType>
        <contentCategory>0x00000002</contentCategory>
        <trustLevel>0x0080</trustLevel>
    </audioInfo>
</app>
<app>
    <appID>0x9012</appID>
    <name>RTP Server</name>
    <providerName>Nokia</providerName>
    <providerURL>http://www.nokia.com</providerURL>
    <description>RTP Audio Server</description>
    <remotingInfo>
        <protocolID>RTP</protocolID>
        <format>99</format>
        <direction>out</direction>
    </remotingInfo>
    <appInfo>
        <appCategory>0xF0000001</appCategory>
        <trustLevel>0x0080</trustLevel>
    </appInfo>
    <audioInfo>
        <audioType>application</audioType>
        <contentCategory>0x02</contentCategory>
    </audioInfo>
    <resourceStatus>free</resourceStatus>
</app>
<app>
    <appID>0x9013</appID>
    <name>Bluetooth A2DP</name>
    <description>Bluetooth A2DP Audio Server</description>
    <remotingInfo>
        <protocolID>BTA2DP</protocolID>
        <direction>out</direction>
    </remotingInfo>
    <audioInfo>
        <audioType>application</audioType>
        <contentCategory>0x02</contentCategory>
    </audioInfo>
    <resourceStatus>free</resourceStatus>
</app>
<app>
    <appID>0x9014</appID>
    <name>Bluetooth HFP</name>
    <description>Bluetooth HFP Audio </description>
    <remotingInfo>
        <protocolID>BTHFP</protocolID>
```

```
            <direction>bi</direction>
        </remotingInfo>
        <audioInfo>
            <audioType>phone</audioType>
            <contentCategory>0x01</contentCategory>
        </audioInfo>
        <resourceStatus>busy</resourceStatus>
    </app>
    <app>
        <appID>0x8011</appID>
        <name>CDB</name>
        <providerName>Nokia</providerName>
        <description>CDB Server Endpoint</description>
        <remotingInfo>
            <protocolID>CDB</protocolID>
            <format>1.1</format>
        </remotingInfo>
    </app>
    <app>
        <appID>0x9016</appID>
        <name>Device Attestation</name>
        <remotingInfo>
            <protocolID>DAP</protocolID>
            <format>1.1</format>
        </remotingInfo>
    </app>
    <Signature Id= "AppListSignature"
    xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
            <CanonicalizationMethod
            Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
            <SignatureMethod Algorithm=
            "http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
            <Reference URI="#mlServerAppList">
                <Transforms>
                    <Transform Algorithm=
                    "http://www.w3.org/2006/12/xml-c14n11"/>
                </Transforms>
                <DigestMethod Algorithm=
                "http://www.w3.org/2000/09/xmldsig#sha1"/>
                <DigestValue>
                    dGhpcyBpcyBub3QgYSB
                    zaWduYXR1cmUK...
                </DigestValue>
            </Reference>
        </SignedInfo>
        <SignatureValue>...</SignatureValue>
    </Signature>
</appList>
```

5.5.5 A_ARG_TYPE_AppCertificateInfo

The value of the non-evented state variable *A_ARG_TYPE_AppCertificateInfo* is an XML block which provides detailed information about the certification status of an application. It is returned in response to a *GetApplicationCertificateInfo* action. The following shows an example XML block of a global CCC drive-certified application.

NOTE: The XML block has been edited. Therefore, the shown *DigestValue* is not correct. The *SignatureValue* has been shortened for readability purpose.

```
<?xml version="1.0" encoding="UTF-8"?>
<certification xml:id="mlAppCertificate">
 <appID>0x00000001</appID>
 <nonce>ODYxMjAwNTgwMDYxOTQxMDkxMzM=</nonce>
 <appUUID>uuid:2fac1234-31f8-11b4-a222-08002b34cff3</appUUID>
 <entity>
  <name>CCC</name>
  <targetList>
   <target></target>
  </targetList>
  <restricted>
   EU,EPE,RUS,CAN,USA,BRA,AMERICA,AUS,KOR,JPN,CHN,HKG,TPE,IND,APAC,AFRICA,WORLD
  </restricted>
  <nonRestricted>
   EU,EPE,RUS,CAN,USA,BRA,AMERICA,AUS,KOR,JPN,CHN,HKG,TPE,IND,APAC,AFRICA,WORLD
```

```
  </nonRestricted>
  <serviceList>
   <service></service>
  </serviceList>
 </entity>
 <properties></properties>
 <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
   <CanonicalizationMethod Algorithm="http://www.w3.org/2006/12/xml-c14n11">
   </CanonicalizationMethod>
   <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1">
   </SignatureMethod>
   <Reference URI="#mlAppCertificate">
    <Transforms>
     <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature">
     </Transform>
     <Transform Algorithm="http://www.w3.org/2006/12/xml-c14n11">
     </Transform>
    </Transforms>
    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod>
    <DigestValue>UbIydFNXUsd89UB+0YpOjy1BUsM=</DigestValue>
   </Reference>
  </SignedInfo>
  <SignatureValue>czTZkj35iKl5+FJH3MMd68uAuF[…]OUw==</SignatureValue>
 </Signature>
</certification>
```

5.6 XML Signature Minimum Set

The MirrorLink Server shall sign the ARG_TYPE_AppListe XML.

The MirrorLink Server shall sign the ARG_TYPE_AppCertificateInfo XML.

Signatures shall follow W3C's recommendation on XML signing, as specified in [3]. The W3C recommendation contains many optional elements for handling the different aspects of the XML signing. In order to reduce the complexity, the following requirements shall be followed for MirrorLink Server and Client devices:

- The **Reference URI** shall not be outside document. It shall refer to the `<appList>` element, which is the parent of the `<ds:Signature>` element, for the UPnP Application List description. It shall refer to the `<certificate>` element, which is the parent of the `<ds:Signature>` element, for the UPnP Application Certification Information description. When the URI attribute is omitted, empty or of an unknown format for the MirrorLink Client to recognize, the MirrorLink Client shall refer to the element listed above.
- MirrorLink Server shall not use XPath or XSLT **XML transformations.**
- The MirrorLink Server shall use **Canonical method** XML version 1.0 (xml-c14n) or 1.1 (xml-c14n11); Canonical XML version 2.0 or later shall not be used.
- The MirrorLink Server shall use SHA-1 **digest method**; other digest methods shall not be used.
- The MirrorLink Server shall use RSA-SHA1 **signature method**; other signature methods, like HMAC-SHA1 or DSA-SHA1, shall not be used.
- The MirrorLink Client shall not use the **KeyInfo** element to identify a public key to verify the signature; it shall use the `applicationPublicKey` element obtained from the DAP `attestationResponse`, for the `TerminalMode:UPnP-Server` component instead.

5.7 Handling of Applications Available via Home Screen Application

A Home Screen application identifies a MirrorLink application, which provides a Home Screen experience, i.e. it allows the consumer to see a list of installed (certified) applications, and allows to launch them. More details are specified in [4].

Classic MirrorLink Mode

In Classic MirrorLink Mode, the MirrorLink Server may provide and advertise a legacy Home Screen application (*application category* "0x00010001"), which should be CCC or Member-drive certified for the connected MirrorLink Client. It's *resourceStatus* shall be set to "free". Applications, which are made available via the legacy Home Screen application, may be advertised with the *resourceStatus* set to "NA". All other applications shall be advertised with the *resourceStatus* set to "free".

A MirrorLink Client may exclude listing applications, which have been advertised with *resourceStatus* "NA" from the MirrorLink Server. It shall list applications, which have been advertised with *resourceStatus* "free" from the MirrorLink Server.

The MirrorLink Client may automatically launch a legacy Home Screen application, whenever a MirrorLink session is established. The MirrorLink Client shall not automatically launch a non-drive certified Home Screen application, while in drive-mode. The MirrorLink Client may automatically launch a non-base certified Home Screen application, while in Park Mode.

In Classic MirrorLink Mode, a MirrorLink Server shall not advertise an Immersive Home Screen application (*application category* "0x00010006").

Immersive MirrorLink Mode

In Immersive MirrorLink Mode, the MirrorLink Server shall provide and advertise an Immersive Home Screen application (*application category* "0x00010006"), which shall be CCC or Member-drive certified for the connected MirrorLink Client. It's *resourceStatus* shall be set to "free". All other remote user interface applications shall be advertised with the *resourceStatus* set to "NA".

In Immersive MirrorLink Mode, a MirrorLink Server shall not advertise a Legacy Home Screen application (*application category* "0x00010001").

A MirrorLink Client shall not list applications, which have been advertised with *resourceStatus* "NA" from the MirrorLink Server. It shall list applications, which have been advertised with *resourceStatus* "free" from the MirrorLink Server.

The MirrorLink Client shall automatically launch the Immersive Home Screen application, whenever a MirrorLink session is established. In case there are multiple Immersive Home Screen appications listed, it is MirrorLink Client dependent, which one will be automatically launched.

5.8 Handling of Application Status Change

Within a MirrorLink session, applications may change their application status dependent on the user operation. New application may come into the foreground, while other applications are moving into the background or are terminating. The MirrorLink Server shall provide notifications of all changes to the MirrorLink Client via the evented *AppStatusUpdate* status variable.

Changing the current foreground applications, e.g. by terminating or moving the current foreground application to the background, may expose uncertified content. In addition to the UPnP *AppStatusUpdate* event, the MirrorLink Server shall additionally update the respective context information, in case the application's user interface is currently projected to the MirrorLink Client, e.g. via VNC or WFD. The MirrorLink Client shall send framebuffer blocking notifications, if the provided application context is not allowed. The MirrorLink Server shall respond to these blocking notification. Details are specified in the respective VNC or WFD specification.

5.9 Handling of different Protocol IDs

The MirrorLink Server shall make all remote user-interface application available via all supported remote protocols, i.e VNC over Wi-Fi, VNC over USB, HSML, WFD and potentially other remote framebuffer transport mechanisms. i.e. an application available via VNC over USB shall be available via WFD as well.

NOTE: This requirement does not apply to UI-less applications, e.g. a stand-alone VNC Server.

The protocol ID in the UPnP application listing and in the URI returned from the UPnP application launch action, shall reflect the framebuffer transport as selected for the MirrorLink session. Protocol identifier of non-remote framebuffer protocols (e.g. CDB, DAP, or RTP), shall not depend on the selected remote framebuffer protocol.

The List of available protocol identifier is defined in Table 4-4.

The MirrorLink Server shall advertise all user-interface applications using the selected remote protocol identifier, as defined in the MirrorLink Core specification. The MirrorLink Server shall update its application listing, if the MirrorLink session switches to another framebuffer transport.

6 XSD Schema

6.1 A_ARG_TYPE_AppStatus XSD Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="urn:schemas-upnp-org:tmapplicationserver:appstatus-1-0"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" id="appstatus">
<xs:element name="appStatusList">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="appStatus" minOccurs="1" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence maxOccurs="unbounded">
      <xs:element name="appID">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="status" minOccurs="1" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="profileID">
          <xs:simpleType>
           <xs:restriction base="xs:nonNegativeInteger">
            <xs:minInclusive value="0"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="statusType">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="Foreground"/>
            <xs:enumeration value="Background"/>
            <xs:enumeration value="Notrunning"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##any" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```

6.2 A_ARG_TYPE_AppList XSD Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="urn:schemas-upnp-org:tmapplicationserver:applist-1-0"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
elementFormDefault="qualified" attributeFormDefault="unqualified" id="applist">
<xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/xmldsig-core-schema.xsd"/>
<xs:element name="appList">
 <xs:complexType>
  <xs:sequence minOccurs="1" maxOccurs="1">
   <xs:element name="app" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="appID" minOccurs="1" maxOccurs="1">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="name" type="xs:string" minOccurs="1"/>
      <xs:element name="variant" type="xs:string" minOccurs="0"/>
      <xs:element name="providerName" type="xs:string" minOccurs="0"/>
      <xs:element name="providerURL" type="xs:string" minOccurs="0"/>
      <xs:element name="description" type="xs:string" minOccurs="0"/>
      <xs:element name="iconList" minOccurs="0">
       <xs:complexType>
        <xs:sequence maxOccurs="unbounded">
         <xs:element name="icon">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="mimetype" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="width" type="xs:positiveInteger" minOccurs="1" maxOccurs="1"/>
            <xs:element name="height" type="xs:positiveInteger" minOccurs="1" maxOccurs="1"/>
            <xs:element name="depth" type="xs:positiveInteger" minOccurs="1" maxOccurs="1"/>
            <xs:element name="url" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
           </xs:sequence>
           <xs:anyAttribute namespace="##any" processContents="lax"/>
          </xs:complexType>
         </xs:element>
         <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="allowedProfileIDs" type="xs:string" minOccurs="0"/>
      <xs:element name="remotingInfo" minOccurs="1" maxOccurs="1">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="protocolID" minOccurs="1" maxOccurs="1">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:pattern value="(VNC|RTP|BTA2DP|BTHFP|DAP|CDB|NONE|WFD|.*-.*)"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="format" type="xs:string" minOccurs="0" maxOccurs="1"/>
         <xs:element name="direction" minOccurs="0" maxOccurs="1" default="out">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="out"/>
            <xs:enumeration value="in"/>
            <xs:enumeration value="bi"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="audioIPL" type="xs:positiveInteger" minOccurs="0" default="4800"/>
         <xs:element name="audioMPL" type="xs:positiveInteger" minOccurs="0" default="9600"/>
         <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="appCertificateURL" type="xs:string" minOccurs="0"/>
      <xs:element name="appInfo" minOccurs="0">
```

```
 <xs:complexType>
  <xs:sequence>
   <xs:element name="appCategory" minOccurs="0" default="0x00000000">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="trustLevel" minOccurs="0" default="0x0000">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="0[Xx][A-Fa-f0-9]{1,4}"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
 </xs:complexType>
</xs:element>
<xs:element name="displayInfo" minOccurs="0">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="contentCategory" minOccurs="0" default="0x00000000">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="contentRules" type="xs:string" minOccurs="0"/>
   <xs:element name="orientation" type="xs:string" minOccurs="0"/>
   <xs:element name="trustLevel" minOccurs="0" default="0x0000">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="0[Xx][A-Fa-f0-9]{1,4}"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
 </xs:complexType>
</xs:element>
<xs:element name="audioInfo" minOccurs="0">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="audioType" minOccurs="1">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:enumeration value="phone"/>
      <xs:enumeration value="application"/>
      <xs:enumeration value="all"/>
      <xs:enumeration value="none"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="contentCategory" minOccurs="1">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="contentRules" type="xs:string" minOccurs="0"/>
   <xs:element name="trustLevel" minOccurs="0" default="0x0000">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="0[Xx][A-Fa-f0-9]{1,4}"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
 </xs:complexType>
</xs:element>
```

```
      <xs:element name="resourceStatus" minOccurs="0" default="free">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:enumeration value="free"/>
         <xs:enumeration value="busy"/>
         <xs:enumeration value="NA"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##any" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:element ref="ds:Signature" minOccurs="1"/>
   <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```

6.3 A_ARG_TYPE_AppCertificateInfo XSD Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="urn:schemas-upnp-org:tmapplicationserver:appstatus-1-0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
elementFormDefault="qualified" attributeFormDefault="unqualified" id=" appcertifcateinfo">
<xs:import schemaLocation="xmldsig-core-schema.xsd"
namespace="http://www.w3.org/2000/09/xmldsig#"/>
<xs:element name="certification">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="appID" minOccurs="1" maxOccurs="1">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="nonce" type="xs:string" minOccurs="1" maxOccurs="1"/>
   <xs:element name="appUUID" type="xs:string" minOccurs="0" maxOccurs="1"/>
   <xs:element name="entity" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="name" type="xs:string" minOccurs="1"/>
      <xs:element name="targetList" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="target" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
         <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="restricted" type="xs:string" minOccurs="1"/>
      <xs:element name="nonRestricted" type="xs:string" minOccurs="1"/>
      <xs:element name="serviceList" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="service" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
         <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##any" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="properties" minOccurs="0" type="xs:string"/>
   <xs:element ref="ds:Signature" minOccurs="1"/>
   <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
  </xs:sequence>
```

```
  <xs:anyAttribute namespace="##any" processContents="lax"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```

7 XML Service Description

```
<?xml version="1.0" encoding="UTF-8"?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
    <specVersion>
        <major>1</major>
        <minor>0</minor>
    </specVersion>
    <actionList>
        <action>
            <name>GetApplicationList</name>
            <argumentList>
                <argument>
                    <name>AppListingFilter</name>
                    <direction>in</direction>
                    <relatedStateVariable>
                        A_ARG_TYPE_String
                    </relatedStateVariable>
                </argument>
                <argument>
                    <name>ProfileID</name>
                    <direction>in</direction>
                    <relatedStateVariable>
                        A_ARG_TYPE_ProfileID
                    </relatedStateVariable>
                </argument>
                <argument>
                    <name>AppListing</name>
                    <direction>out</direction>
                    <relatedStateVariable>
                        A_ARG_TYPE_AppList
                    </relatedStateVariable>
                </argument>
            </argumentList>
        </action>
        <action>
            <name>LaunchApplication</name>
            <argumentList>
                <argument>
                    <name>AppID</name>
                    <direction>in</direction>
                    <relatedStateVariable>
                        A_ARG_TYPE_AppID
                    </relatedStateVariable>
                </argument>
                <argument>
                    <name>ProfileID</name>
                    <direction>in</direction>
                    <relatedStateVariable>
                        A_ARG_TYPE_ProfileID
                    </relatedStateVariable>
                </argument>
                <argument>
                    <name>AppURI</name>
                    <direction>out</direction>
                    <relatedStateVariable>
                        A_ARG_TYPE_URI
                    </relatedStateVariable>
                </argument>
            </argumentList>
        </action>
        <action>
            <name>TerminateApplication</name>
            <argumentList>
                <argument>
                    <name>AppID</name>
                    <direction>in</direction>
                    <relatedStateVariable>
                        A_ARG_TYPE_AppID
                    </relatedStateVariable>
```

```
        </argument>
        <argument>
            <name>ProfileID</name>
            <direction>in</direction>
            <relatedStateVariable>
                A_ARG_TYPE_ProfileID
            </relatedStateVariable>
        </argument>
        <argument>
            <name>TerminationResult</name>
            <direction>out</direction>
            <relatedStateVariable>
                A_ARG_TYPE_Bool
            </relatedStateVariable>
        </argument>
    </argumentList>
</action>
<action>
    <name>GetApplicationStatus</name>
    <argumentList>
        <argument>
            <name>AppID</name>
            <direction>in</direction>
            <relatedStateVariable>
                A_ARG_TYPE_AppID
            </relatedStateVariable>
        </argument>
        <argument>
            <name>AppStatus</name>
            <direction>out</direction>
            <relatedStateVariable>
                A_ARG_TYPE_AppStatus
            </relatedStateVariable>
        </argument>
    </argumentList>
</action>
<action>
    <name>GetApplicationCertificateInfo</name>
    <argumentList>
        <argument>
            <name>AppID</name>
            <direction>in</direction>
            <relatedStateVariable>
                A_ARG_TYPE_AppID
            </relatedStateVariable>
        </argument>
        <argument>
            <name>AppCertification</name>
            <direction>out</direction>
            <relatedStateVariable>
                A_ARG_TYPE_AppCertificateInfo
            </relatedStateVariable>
        </argument>
    </argumentList>
</action>
<action>
    <name>GetCertifiedApplicationsList</name>
    <argumentList>
        <argument>
            <name>AppCertFilter</name>
            <direction>in</direction>
            <relatedStateVariable>
                A_ARG_TYPE_String
            </relatedStateVariable>
        </argument>
        <argument>
            <name>ProfileID</name>
            <direction>in</direction>
            <relatedStateVariable>
                A_ARG_TYPE_ProfileID
            </relatedStateVariable>
        </argument>
        <argument>
            <name>CertifiedAppList</name>
            <direction>out</direction>
            <relatedStateVariable>
                A_ARG_TYPE_String
            </relatedStateVariable>
```

```
            </argument>
        </argumentList>
    </action>
    <action>
        <name>GetAppCertificationStatus</name>
        <argumentList>
            <argument>
                <name>AppID</name>
                <direction>in</direction>
                <relatedStateVariable>
                    A_ARG_TYPE_AppID
                </relatedStateVariable>
            </argument>
            <argument>
                <name>AppCertFilter</name>
                <direction>in</direction>
                <relatedStateVariable>
                    A_ARG_TYPE_String
                </relatedStateVariable>
            </argument>
            <argument>
                <name>ProfileID</name>
                <direction>in</direction>
                <relatedStateVariable>
                    A_ARG_TYPE_ProfileID
                </relatedStateVariable>
            </argument>
            <argument>
                <name>AppCertified</name>
                <direction>out</direction>
                <relatedStateVariable>
                    A_ARG_TYPE_Bool
                </relatedStateVariable>
            </argument>
        </argumentList>
    </action>
    <action>
        <name>SetAllowedApplicationsList</name>
        <argumentList>
            <argument>
                <name>AllowedAppListNonRestricted</name>
                <direction>in</direction>
                <relatedStateVariable>
                    A_ARG_TYPE_String
                </relatedStateVariable>
            </argument>
            <argument>
                <name>AllowedAppListRestricted </name>
                <direction>in</direction>
                <relatedStateVariable>
                    A_ARG_TYPE_String
                </relatedStateVariable>
            </argument>
            <argument>
                <name>ProfileID</name>
                <direction>in</direction>
                <relatedStateVariable>
                    A_ARG_TYPE_ProfileID
                </relatedStateVariable>
            </argument>
        </argumentList>
    </action>
</actionList>
<serviceStateTable>
    <stateVariable sendEvents="yes">
        <name>AppStatusUpdate</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="yes">
        <name>AppListUpdate</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>A_ARG_TYPE_AppStatus</name>
        <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
        <name>A_ARG_TYPE_AppList</name>
```

```
            <dataType>string</dataType>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>A_ARG_TYPE_AppID</name>
            <dataType>string</dataType>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>A_ARG_TYPE_ProfileID</name>
            <dataType>ui4</dataType>
            <defaultValue>0</defaultValue>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>A_ARG_TYPE_URI</name>
            <dataType>uri</dataType>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>A_ARG_TYPE_String</name>
            <dataType>string</dataType>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>A_ARG_TYPE_Bool</name>
            <dataType>string</dataType>
            <defaultValue>false</defaultValue>
            <allowedValueList>
                <allowedValue>false</allowedValue>
                <allowedValue>true</allowedValue>
            </allowedValueList>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>A_ARG_TYPE_INT</name>
            <dataType>ui4</dataType>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>A_ARG_TYPE_AppCertificateInfo</name>
            <dataType>string</dataType>
        </stateVariable>
    </serviceStateTable>
</scpd>
```

Annex A (normative): Application Context Information

A.1 Trust Level

Trust level used in the Context Information Pseudo Encoding is given in Table A-1.

Table A-1: Trust Level

Value	Description	Trust Level
0x0000	Unknown	No Trust
0x0040	User Configuration	Trust the user The provided data is under control of the user. The user is either directly setting the values or using provided default settings.
0x0060	Self-Registered Application	Trust the application The provided data is under control of the application. The values are defined from the application developer and provided to the VNC and UPnP MirrorLink server via a specific API.
0x0080	Registered Application	Trust the VNC and UPnP MirrorLink server The provided data is under sole control of the VNC and UPnP MirrorLink server. The application is known to them and shall be uniquely identified. The user and the application shall not be able to change the values.
0x00A0	Application certificate	Trust a 3rd party certification entity The provided data is under sole control of the VNC and UPnP MirrorLink server. The data is derived from a valid application certificate. The VNC and UPnP MirrorLink server shall not change the provided data. The user shall not be able to change the values. The application shall not be able to change the values, other than through the MirrorLink API.

NOTE: The structure of the application certificate, the certification provider and the certification process are not subject to the present document.

A.2 Application Categories

The application categories and sub-categories are given in Table A-2. The table can be amended in future versions of the present document.

Table A-2: Application Categories

Application Category	Application Sub-category	Description
0x0000	0x0000	Unknown Application
0x0001	0x0000	General UI Framework
	0x0001	Home screen / Start-up Screen This application category shall describe the phone's start-up screen, e.g. shown after booting or after switching to the automotive context.
	0x0002	Menu
	0x0003	Notification
	0x0004	Application listing
	0x0005	Settings

Application Category	Application Sub-category	Description
	0x0006	Immersive Home Screen
0x0002	0x0000	General Phone call application
	0x0001	Contact list
	0x0002	Call log
	0x0003	Immersive Phone Call
0x0003	0x0000	General Media applications
	0x0001	Music
	0x0002	Video
	0x0003	Gaming
	0x0004	Image
0x0004	0x0000	General Messaging applications
	0x0001	SMS
	0x0002	MMS
	0x0003	Email
0x0005	0x0000	General Navigation
0x0006	0x0000	General Browser
	0x0001	Application Store
0x0007	0x0000	General Productivity
	0x0001	Document Viewer
	0x0002	Document Editor
0x0008	0x0000	General Information
	0x0001	News
	0x0002	Weather
	0x0003	Stocks
	0x0004	Travel
	0x0005	Sports
	0x0006	Clock
0x0009	0x0000	General Social Networking
0x000A	0x0000	General Personal Information Management
	0x0001	Calendar
	0x0002	Notes
0xF000	0x0000	General UI less applications
	0x0001	Server functionality (use *protocolID* to identify the server protocol)
	0x0002	Client functionality (use *protocolID* to identify the client protocol)
	0x0010	Voice Command Engine (shall only be used in RTP header extensions)
	0x0020	Conversational Audio (shall only be used in RTP header extensions)
	0xFFFF	Switch to MirrorLink Client native UI
0xFFFE	0x0000 – 0xFFFF	Testing & Certification The use of this category will be specified in a separate “Testing and Certification” document.

Application Category	Application Sub-category	Description
0xFFFF	0x0000	General System
	0x0001	PIN input for Device unlock, Device Unlock Notification
	0x0002	Bluetooth PIN code Input
	0x000F	Other password input
	0x0010	Voice Command Confirmation

NOTE: The p*rotocolID* is provided in response to a *GetApplicationList* action within *TmApplicationServer*:1 service.

A.3 Content Categories

The content categories are represented as 32-bit unsigned integer values.

Table A-3 lists content categories for visual content of an application and are specified as values of the *contentCategory* child element of the *displayInfo* element in the *A_ARG_TYPE_AppList* state variable for the *TmApplicationServer*:1 service.

Table A-3: Content Categories for Visual Content

Content Category (Bit)	Description
0	Text
1	Video
2	Image
3	Vector Graphics
4	3D Graphics
5	User Interface (e.g. Application menu)
6..15	Reserved for future use
16	Car Mode (Application user interface is complying with all rules for a restricted driving mode)
17..30	Reserved for future use
31	Miscellaneous content

Table A-4 lists content categories for audio content of an application from the MirrorLink Server's perspective and are specified as values of the *contentCategory* child element of the *audioInfo* element in the *A_ARG_TYPE_AppList* state variable for the *TmApplicationServer*:1 service.

Table A-4: Content Categories for Audio Content

Content Category (Bit)	Description
0	Phone Audio
1	Media Audio Out
2	Media Audio In
3	Voice Command Out
4	Voice Command In
5..30	Reserved for future use
31	Miscellaneous content

If no bit is set, then the content category of that application shall be treated as "Unknown".

Annex B (informative): Authors and Contributors

The following people have contributed to the present document:

Rapporteur:	Dr. Jörg Brakensiek, E-Qualus (for Car Connectivity Consortium LLC)
Other contributors:	Matthias Benesch, Daimler AG
	Raja Bose, Nokia Corporation
	Dennis Fernahl, Carmeq (for Volkswagen AG)
	Christopher Seubert, Carmeq (for Volkswagen AG)

History

Document history		
V1.3.0	October 2017	Publication
V1.3.1	October 2019	Publication