INTERNATIONAL STANDARD

ISO/IEC 29341-20-1

First edition
2017-09

Information technology — UPnP Device Architecture —

Part 20-1: Audio video device control protocol — Level 4 — Audio video architecture

Technologies de l'information — Architecture de dispositif UPnP —

Partie 20-1: Protocole de contrôle de dispositif audio-vidéo — Niveau 4 — Architecture audio-vidéo

Reference number
ISO/IEC 29341-20-1:2017(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

CONTENTS

List of Figures

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see http://www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of Standard, the meaning of the ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword – Supplementary information

ISO/IEC 29341-20-1 was prepared by UPnP Forum and adopted, under the PAS procedure, by joint technical committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

The list of all currently available parts of ISO/IEC 29341 series, under the general title *Information technology — UPnP Device Architecture*, can be found on the ISO web site.

Introduction

ISO and IEC draw attention to the fact that it is claimed that compliance with this document may involve the use of patents as indicated below.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights. The holders of -these patent rights have assured ISO and IEC that they are willing to negotiate licenses under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC.

Intel Corporation has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Intel Corporation
Standards Licensing Department
5200 NE Elam Young Parkway
MS: JFS-98
USA – Hillsboro, Oregon 97124

Microsoft Corporation has informed IEC and ISO that it has patent applications or granted patents as listed below:

6101499 / US; 6687755 / US; 6910068 / US; 7130895 / US; 6725281 / US; 7089307 / US; 7069312 / US; 10/783 524 /US

Information may be obtained from:

Microsoft Corporation
One Microsoft Way
USA – Redmond WA 98052

Philips International B.V. has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Philips International B.V. – IP&S
High Tech campus, building 44 3A21
NL – 5656 Eindhoven

NXP B.V. (NL) has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

NXP B.V. (NL)
High Tech campus 60
NL – 5656 AG Eindhoven

Matsushita Electric Industrial Co. Ltd. has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Matsushita Electric Industrial Co. Ltd.
1-3-7 Shiromi, Chuoh-ku
JP – Osaka 540-6139

Hewlett Packard Company has informed IEC and ISO that it has patent applications or granted patents as listed below:

5 956 487 / US; 6 170 007 / US; 6 139 177 / US; 6 529 936 / US; 6 470 339 / US; 6 571 388 / US; 6 205 466 / US

Information may be obtained from:

Hewlett Packard Company
1501 Page Mill Road
USA – Palo Alto, CA 94304

Samsung Electronics Co. Ltd. has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Digital Media Business, Samsung Electronics Co. Ltd.
416 Maetan-3 Dong, Yeongtang-Gu,
KR – Suwon City 443-742

Huawei Technologies Co., Ltd. has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Huawei Technologies Co., Ltd.
Administration Building, Bantian Longgang District
Shenzhen – China 518129

Qualcomm Incorporated has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Qualcomm Incorporated
5775 Morehouse Drive
San Diego, CA – USA 92121

Telecom Italia S.p.A.has informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Telecom Italia S.p.A.
Via Reiss Romoli, 274
Turin - Italy 10148

Cisco Systems informed IEC and ISO that it has patent applications or granted patents.

Information may be obtained from:

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA – USA 95134

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Original UPnP Document

Reference may be made in this document to original UPnP documents. These references are retained in order to maintain consistency between the specifications as published by ISO/IEC and by UPnP Implementers Corporation and later by UPnP Forum. The following table indicates the original UPnP document titles and the corresponding part of ISO/IEC 29341:

UPnP Document Title	ISO/IEC 29341 Part
UPnP Device Architecture 1.0	ISO/IEC 29341-1:2008
UPnP Device Architecture Version 1.0	ISO/IEC 29341-1:2011
UPnP Device Architecture 1.1	ISO/IEC 29341-1-1:2011
UPnP Device Architecture 2.0	ISO/IEC 29341-1-2
UPnP Basic:1 Device	ISO/IEC 29341-2
UPnP AV Architecture:1	ISO/IEC 29341-3-1:2008
UPnP AV Architecture:1	ISO/IEC 29341-3-1:2011
UPnP AVTransport:1 Service	ISO/IEC 29341-3-10
UPnP ConnectionManager:1 Service	ISO/IEC 29341-3-11
UPnP ContentDirectory:1 Service	ISO/IEC 29341-3-12
UPnP RenderingControl:1 Service	ISO/IEC 29341-3-13
UPnP MediaRenderer:1 Device	ISO/IEC 29341-3-2
UPnP MediaRenderer:2 Device	ISO/IEC 29341-3-2:2011
UPnP MediaServer:1 Device	ISO/IEC 29341-3-3
UPnP AVTransport:2 Service	ISO/IEC 29341-4-10:2008
UPnP AVTransport:2 Service	ISO/IEC 29341-4-10:2011
UPnP ConnectionManager:2 Service	ISO/IEC 29341-4-11:2008
UPnP ConnectionManager:2 Service	ISO/IEC 29341-4-11:2011
UPnP ContentDirectory:2 Service	ISO/IEC 29341-4-12
UPnP RenderingControl:2 Service	ISO/IEC 29341-4-13:2008
UPnP RenderingControl:2 Service	ISO/IEC 29341-4-13:2011
UPnP ScheduledRecording:1	ISO/IEC 29341-4-14
UPnP ScheduledRecording:2	ISO/IEC 29341-4-14:2011
UPnP MediaRenderer:2 Device	ISO/IEC 29341-4-2
UPnP MediaServer:2 Device	ISO/IEC 29341-4-3
UPnP AV Datastructure Template:1	ISO/IEC 29341-4-4:2008
UPnP AV Datastructure Template:1	ISO/IEC 29341-4-4:2011
UPnP DigitalSecurityCamera:1 Device	ISO/IEC 29341-5-1
UPnP DigitalSecurityCameraMotionImage:1 Service	ISO/IEC 29341-5-10
UPnP DigitalSecurityCameraSettings:1 Service	ISO/IEC 29341-5-11
UPnP DigitalSecurityCameraStillImage:1 Service	ISO/IEC 29341-5-12
UPnP HVAC_System:1 Device	ISO/IEC 29341-6-1
UPnP ControlValve:1 Service	ISO/IEC 29341-6-10
UPnP HVAC_FanOperatingMode:1 Service	ISO/IEC 29341-6-11
UPnP FanSpeed:1 Service	ISO/IEC 29341-6-12
UPnP HouseStatus:1 Service	ISO/IEC 29341-6-13
UPnP HVAC_SetpointSchedule:1 Service	ISO/IEC 29341-6-14
UPnP TemperatureSensor:1 Service	ISO/IEC 29341-6-15
UPnP TemperatureSetpoint:1 Service	ISO/IEC 29341-6-16
UPnP HVAC_UserOperatingMode:1 Service	ISO/IEC 29341-6-17

UPnP HVAC_ZoneThermostat:1 Device	ISO/IEC 29341-6-2
UPnP BinaryLight:1 Device	ISO/IEC 29341-7-1
UPnP Dimming:1 Service	ISO/IEC 29341-7-10
UPnP SwitchPower:1 Service	ISO/IEC 29341-7-11
UPnP DimmableLight:1 Device	ISO/IEC 29341-7-2
UPnP InternetGatewayDevice:1 Device	ISO/IEC 29341-8-1
UPnP LANHostConfigManagement:1 Service	ISO/IEC 29341-8-10
UPnP Layer3Forwarding:1 Service	ISO/IEC 29341-8-11
UPnP LinkAuthentication:1 Service	ISO/IEC 29341-8-12
UPnP RadiusClient:1 Service	ISO/IEC 29341-8-13
UPnP WANCableLinkConfig:1 Service	ISO/IEC 29341-8-14
UPnP WANCommonInterfaceConfig:1 Service	ISO/IEC 29341-8-15
UPnP WANDSLLinkConfig:1 Service	ISO/IEC 29341-8-16
UPnP WANEthernetLinkConfig:1 Service	ISO/IEC 29341-8-17
UPnP WANIPConnection:1 Service	ISO/IEC 29341-8-18
UPnP WANPOTSLinkConfig:1 Service	ISO/IEC 29341-8-19
UPnP LANDevice:1 Device	ISO/IEC 29341-8-2
UPnP WANPPPConnection:1 Service	ISO/IEC 29341-8-20
UPnP WLANConfiguration:1 Service	ISO/IEC 29341-8-21
UPnP WANDevice:1 Device	ISO/IEC 29341-8-3
UPnP WANConnectionDevice:1 Device	ISO/IEC 29341-8-4
UPnP WLANAccessPointDevice:1 Device	ISO/IEC 29341-8-5
UPnP Printer:1 Device	ISO/IEC 29341-9-1
UPnP ExternalActivity:1 Service	ISO/IEC 29341-9-10
UPnP Feeder:1.0 Service	ISO/IEC 29341-9-11
UPnP PrintBasic:1 Service	ISO/IEC 29341-9-12
UPnP Scan:1 Service	ISO/IEC 29341-9-13
UPnP Scanner:1.0 Device	ISO/IEC 29341-9-2
UPnP QoS Architecture:1.0	ISO/IEC 29341-10-1
UPnP QosDevice:1 Service	ISO/IEC 29341-10-10
UPnP QosManager:1 Service	ISO/IEC 29341-10-11
UPnP QosPolicyHolder:1 Service	ISO/IEC 29341-10-12
UPnP QoS Architecture:2	ISO/IEC 29341-11-1
UPnP QosDevice:2 Service	ISO/IEC 29341-11-10
UPnP QosManager:2 Service	ISO/IEC 29341-11-11
UPnP QosPolicyHolder:2 Service	ISO/IEC 29341-11-12
UPnP QOS v2 Schema Files	ISO/IEC 29341-11-2
UPnP RemoteUIClientDevice:1 Device	ISO/IEC 29341-12-1
UPnP RemoteUIClient:1 Service	ISO/IEC 29341-12-10
UPnP RemoteUIServer:1 Service	ISO/IEC 29341-12-11
UPnP RemoteUIServerDevice:1 Device	ISO/IEC 29341-12-2
UPnP DeviceSecurity:1 Service	ISO/IEC 29341-13-10
UPnP SecurityConsole:1 Service	ISO/IEC 29341-13-11
UPnP ContentDirectory:3 Service	ISO/IEC 29341-14-12:2011
UPnP MediaServer:3 Device	ISO/IEC 29341-14-3:2011
UPnP ContentSync:1	ISO/IEC 29341-15-10:2011
UPnP Low Power Architecture:1	ISO/IEC 29341-16-1:2011

UPnP LowPowerProxy:1 Service	ISO/IEC 29341-16-10:2011
UPnP LowPowerDevice:1 Service	ISO/IEC 29341-16-11:2011
UPnP QoS Architecture:3	ISO/IEC 29341-17-1:2011
UPnP QosDevice:3 Service	ISO/IEC 29341-17-10:2011
UPnP QosManager:3 Service	ISO/IEC 29341-17-11:2011
UPnP QosPolicyHolder:3 Service	ISO/IEC 29341-17-12:2011
UPnP QosDevice:3 Addendum	ISO/IEC 29341-17-13:2011
UPnP RemoteAccessArchitecture:1	ISO/IEC 29341-18-1:2011
UPnP InboundConnectionConfig:1 Service	ISO/IEC 29341-18-10:2011
UPnP RADAConfig:1 Service	ISO/IEC 29341-18-11:2011
UPnP RADASync:1 Service	ISO/IEC 29341-18-12:2011
UPnP RATAConfig:1 Service	ISO/IEC 29341-18-13:2011
UPnP RAClient:1 Device	ISO/IEC 29341-18-2:2011
UPnP RAServer:1 Device	ISO/IEC 29341-18-3:2011
UPnP RADiscoveryAgent:1 Device	ISO/IEC 29341-18-4:2011
UPnP SolarProtectionBlind:1 Device	ISO/IEC 29341-19-1:2011
UPnP TwoWayMotionMotor:1 Service	ISO/IEC 29341-19-10:2011
UPnP AV Architecture:2	ISO/IEC 29341-20-1
UPnP AVTransport:3 Service	ISO/IEC 29341-20-10
UPnP ConnectionManager:3 Service	ISO/IEC 29341-20-11
UPnP ContentDirectory:4 Device	ISO/IEC 29341-20-12
UPnP RenderingControl:3 Service	ISO/IEC 29341-20-13
UPnP ScheduledRecording:2 Service	ISO/IEC 29341-20-14
UPnP MediaRenderer:3 Service	ISO/IEC 29341-20-2
UPnP MediaServer:4 Device	ISO/IEC 29341-20-3
UPnP AV Datastructure Template:1	ISO/IEC 29341-20-4
UPnP InternetGatewayDevice:2 Device	ISO/IEC 29341-24-1
UPnP WANIPConnection:2 Service	ISO/IEC 29341-24-10
UPnP WANIPv6FirewallControl:1 Service	ISO/IEC 29341-24-11
UPnP WANConnectionDevice:2 Service	ISO/IEC 29341-24-2
UPnP WANDevice:2 Device	ISO/IEC 29341-24-3
UPnP Telephony Architecture:2	ISO/IEC 29341-26-1
UPnP CallManagement:2 Service	ISO/IEC 29341-26-10
UPnP MediaManagement:2 Service	ISO/IEC 29341-26-11
UPnP Messaging:2 Service	ISO/IEC 29341-26-12
UPnP PhoneManagement:2 Service	ISO/IEC 29341-26-13
UPnP AddressBook:1 Service	ISO/IEC 29341-26-14
UPnP Calendar:1 Service	ISO/IEC 29341-26-15
UPnP Presense:1 Service	ISO/IEC 29341-26-16
UPnP TelephonyClient:2 Device	ISO/IEC 29341-26-2
UPnP TelephonyServer:2 Device	ISO/IEC 29341-26-3
UPnP Friendly Info Update:1 Service	ISO/IEC 29341-27-1
UPnP MultiScreen MultiScreen Architecture:1	ISO/IEC 29341-28-1
UPnP MultiScreen Application Management:1 Service	ISO/IEC 29341-28-10
UPnP MultiScreen Screen:1 Device	ISO/IEC 29341-28-2
UPnP MultiScreen Application Management:2 Service	ISO/IEC 29341-29-10
UPnP MultiScreen Screen:2 Device	ISO/IEC 29341-29-2

UPnP IoT Management and Control Architecture Overview:1	ISO/IEC 29341-30-1
UPnP DataStore:1 Service	ISO/IEC 29341-30-10
UPnP IoT Management and Control Data Model:1 Service	ISO/IEC 29341-30-11
UPnP IoT Management and Control Transport Generic:1 Service	ISO/IEC 29341-30-12
UPnP IoT Management and Control:1 Device	ISO/IEC 29341-30-2
UPnP Energy Management:1 Service	ISO/IEC 29341-31-1

1 Scope

1.1 Introductioin

This document describes the overall UPnP AV Architecture, which forms the foundation for the UPnP AV Device and Service templates. The AV Architecture defines the general interaction between UPnP control points and UPnP AV devices. It is independent of any particular device type, content format, and transfer protocol. It supports a variety of devices such as TVs, VCRs, CD/DVD players/jukeboxes, settop boxes, stereos systems, MP3 players, still-image cameras, camcorders, electronic picture frames (EPFs), and the PC. The AV Architecture allows devices to support different types of formats for the entertainment content (such as MPEG2, MPEG4, JPEG, MP3, Windows Media Architecture (WMA), bitmaps (BMP), NTSC, PAL, ATSC, etc.) and multiple types of transfer protocols (such as IEC-61883/IEEE-1394, HTTP GET, RTP, HTTP PUT/POST, TCP/IP, etc.). The following clauses describe the AV Architecture and how the various UPnP AV devices and services work together to enable various end-user scenarios.

1.2 Goals

The UPnP AV Architecture was explicitly defined to meet the following goals:

- To support arbitrary transfer protocols and content formats.
- To enable the AV content to flow directly between devices without any intervention from the control point.
- To enable control points to remain independent of any particular transfer protocol and content format. This allows control points to transparently support new protocols and formats.
- Scalability, i.e. support of devices with very low resources, especially memory and processing power as well as full-featured devices.
- Synchronized playback to multiple rendering devices.
- Access Control, Content Protection, and Digital Rights Management.

1.3 Non-Goals

The UPnP AV Architecture does not enable any of the following:

- Two-way Interactive Communication, such as audio and video conferencing, Internet gaming, etc.

2 Normative references

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

[1] *AVTransport:3*, UPnP Forum, December, 31, 2010.
Available at: http://www.upnp.org/specs/av/UPnP-av-AVTransport-v3-Service-20101231.pdf.
Latest version available at: http://www.upnp.org/specs/av/UPnP-av-AVTransport-v3-Service.pdf.

[2] *ContentDirectory:4*, UPnP Forum, December, 31, 2010.
Available at: http://www.upnp.org/specs/av/UPnP-av-ContentDirectory-v4-Service-20101231.pdf.
Latest version available at: http://www.upnp.org/specs/av/UPnP-av-ContentDirectory-v4-Service.pdf.

[3] *ConnectionManager:3*, UPnP Forum, December, 31, 2010.
Available at: http://www.upnp.org/specs/av/UPnP-av-ConnectionManager-v2-Service-20101231.pdf.
Latest version available at: http://www.upnp.org/specs/av/UPnP-av-ConnectionManager-v3-Service.pdf.

[4] *MediaRenderer:3*, UPnP Forum, December, 31, 2010.
Available at: http://www.upnp.org/specs/av /UPnP-av-MediaRenderer-v3-Device-20101231.pdf.
Latest version available at: http://www.upnp.org/specs/av /UPnP-av-MediaRenderer-v3-Device.pdf.

[5] *MediaServer:4*, UPnP Forum, December, 31, 2010.
Available at: http://www.upnp.org/specs/av/UPnP-av-MediaServer-v4-Device-20101231.pdf.
Latest version available at: http://www.upnp.org/specs/av/UPnP-av-MediaServer-v4-Device.pdf.

[6] *RenderingControl:3*, UPnP Forum, December, 31, 2010.
Available at: http://www.upnp.org/specs/av/UPnP-av-RenderingControl-v3-Service-20101231.pdf.
Latest version available at: http://www.upnp.org/specs/av/UPnP-av-RenderingControl-v3-Service.pdf.

[7] *ScheduledRecording:2*, UPnP Forum, December, 31, 2010.
Available at: http://www.upnp.org/specs/av/UPnP-av-ScheduledRecording-v2-Service-20101231.pdf.
Latest version available at: http://www.upnp.org/specs/av/UPnP-av-ScheduledRecording-v2-Service.pdf.

3 Terms, definitions, symbols and abbreviations

For the purposes of this document, the terms and definitions given in [1] and the following apply:

3.1
AVT
AVTransport Service

3.2
CDS
ContentDirectory Service

3.3
CM
ConnectionManager Service

3.4
MR
MediaRenderer Device

3.5
MS
MediaServer Device

3.6
RCS
RenderingControl Service

3.7
SRS
ScheduledRecording Service

3.8
2-box model
Denotes an interaction model between 1 control point and 1 UPnP device.

3.9
3-box model
Denotes an interaction model between 1control point and 2 UPnP devices, where the 2 UPnP devices are used in different roles to obtain the described feature.

4 Architectural Overview

In most (non-AV) UPnP scenarios, a control point controls the operation of one or more UPnP devices in order to accomplish the desired behavior. Although the control point is managing multiple devices, all interactions occur in isolation between the control point and each device. The control point coordinates the operation of each device to achieve an overall, synchronized, end-user effect. The individual devices do not interact directly with each another. All of the coordination between the devices is performed by the control point and not the devices themselves.

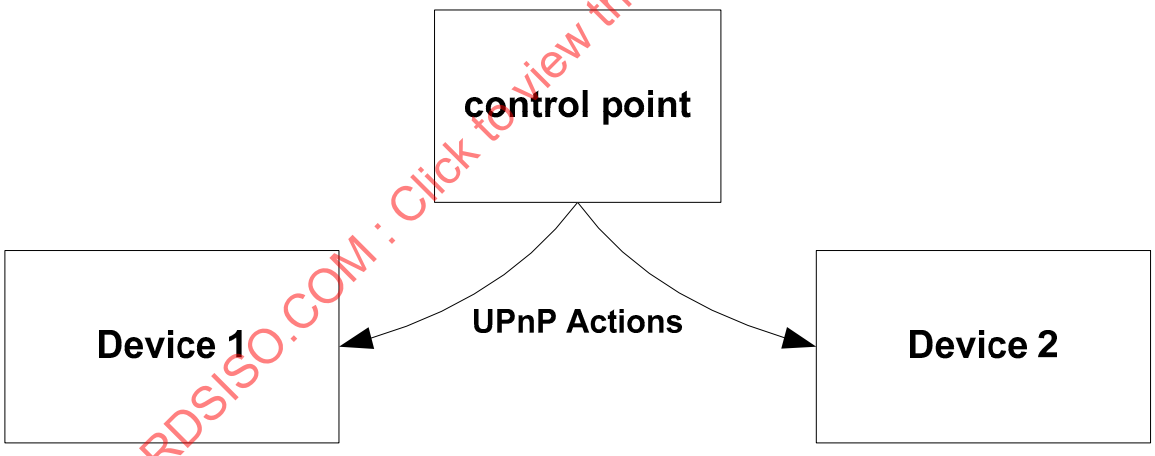

Figure 1 — Typical UPnP Device Interaction Model

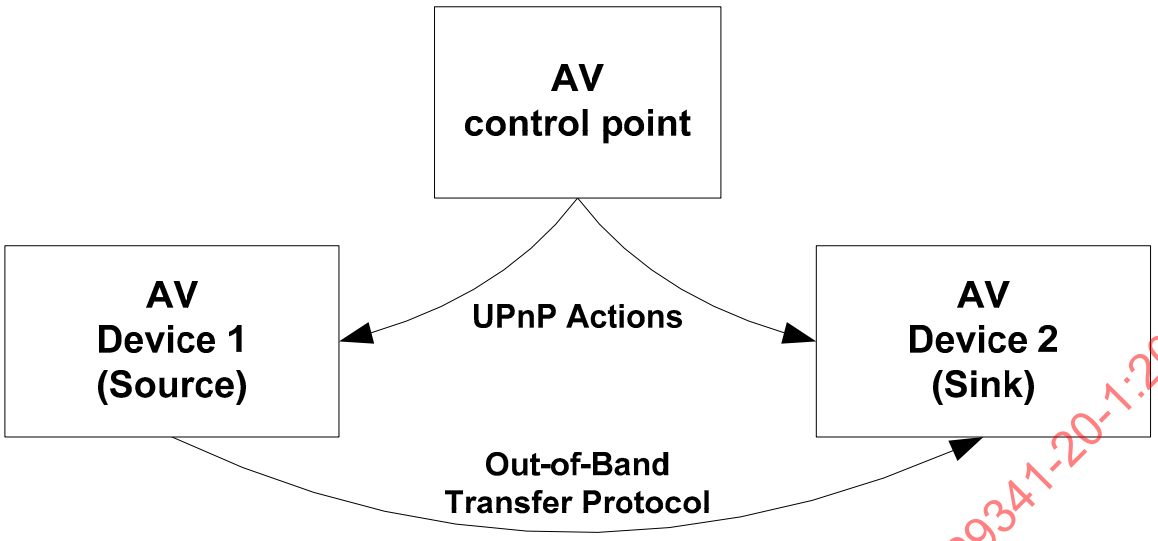

Figure 2 — UPnP AV Device Interaction Model

Most AV scenarios involve the flow of (entertainment) content (i.e. a movie, song, picture, etc.) from one device to another. As shown in Figure 2, an AV control point interacts with two or more UPnP devices acting as source and sink, respectively. Although the control point coordinates and synchronizes the behavior of both devices, the devices themselves interact with each other using a non-UPnP ("out-of-band") communication protocol. The control point uses UPnP to initialize and configure both devices so that the desired content is transferred from one device to the other. However, since the content is transferred using an "out-of-band" transfer protocol, the control point is not directly involved in the actual transfer of the content. The control point configures the devices as needed, triggers the flow of content, then gets out of the way. Thus, after the transfer has begun, the control point can be disconnected without disrupting the flow of content. In other words, the core task (i.e. transferring the content) continues to function even without the control point present.

As described in the above scenario, three distinct entities are involved: the control point, the source of the media content (called the "MediaServer"), and the sink for the content (called the "MediaRenderer"). Throughout the remainder of the document, all three entities are described as if they were independent devices on the network. Although this configuration may be common (i.e. a remote control, a VCR, and a TV), the AV Architecture supports arbitrary combinations of these entities within a single physical device. For example, a TV can be treated as a rendering device (e.g. a display). However, since most TVs contain a built-in tuner, the TV can also act as a server device because it could tune to a particular channel and send that content to a MediaRenderer [4] (e.g. its local display or some remote device such as a tuner-less display). Similarly, many MediaServers and/or MediaRenderers may also include control point functionality. For example, an MP3 Renderer will likely have some UI controls (e.g. a small display and some buttons) that allow the user to control the playback of music.

5 Playback Architecture

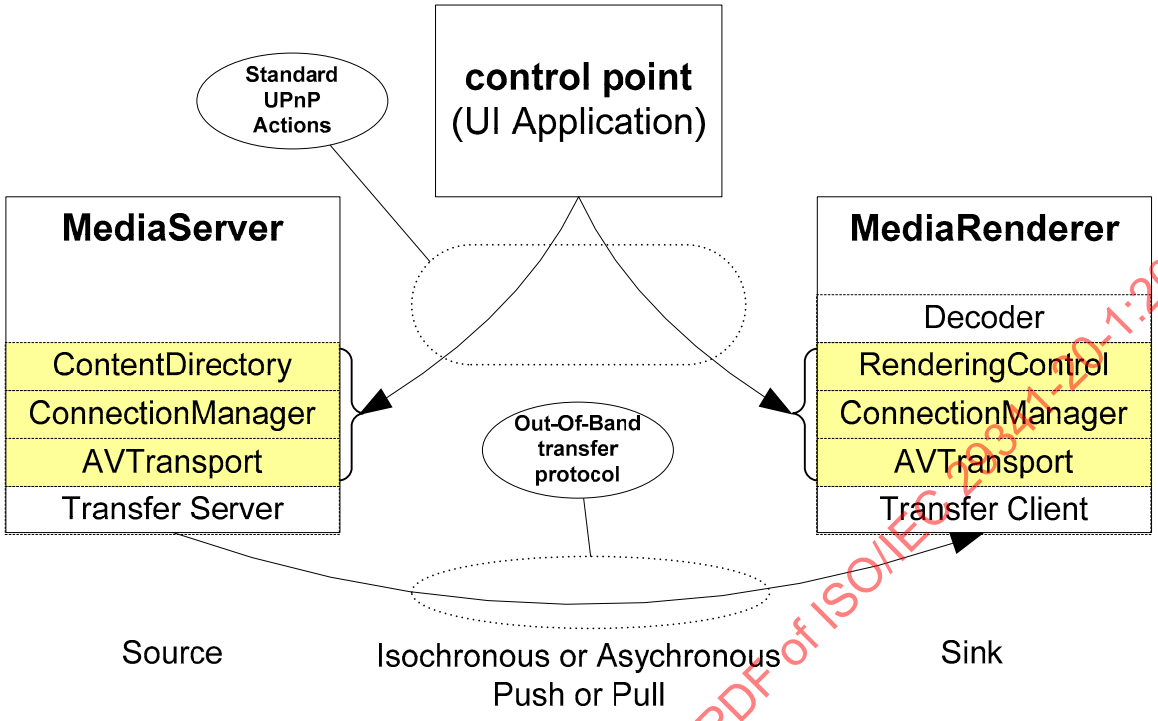

Figure 3 — General Device Architecture aka the 3-Box model

The most common task that end-users want to perform is to render (i.e. play) individual items of content on a specific rendering device. As shown in Figure 3, the content playback scenario involves three distinct UPnP components: a MediaServer [5], a MediaRenderer, and a UPnP control point. These three components (each with a well-defined role) work together to accomplish the task. In this scenario, the MediaServer contains (entertainment) content that the user wants to render (e.g. display or listen to) on the MediaRenderer. The user interacts with the control point's UI to locate and select the desired content on the MediaServer and to select the target MediaRenderer.

The MediaServer contains or has access to a variety of entertainment content, either stored locally or stored on an external device that is accessible via the MediaServer. The MediaServer is able to access its content and transmit it to another device via the network using some type of transfer protocol. The content exposed by the MediaServer may include arbitrary types of content including video, audio, and/or still images. The content is transmitted over the network using a transfer protocol and data format that is that is understood by the MediaServer and MediaRenderer. MediaServers may support one or multiple transfer protocols and data formats for each content item or be able to convert the format of a given content item into another formats on the fly. Examples of a MediaServer include a VCR, CD/DVD player/jukebox, smartphone, camera, camcorder, PC, set-top box, satellite receiver, audio tape player, etc.

The MediaRenderer obtains content from a MediaServer via network. Examples of a MediaRenderer include TV, stereo, network-enabled speakers, MP3 players, Electronic Picture Frame (EPF), a music-controlled water fountain, etc.. The type of content that a MediaRenderer can receive depends on the transfer protocols and data formats that it supports. Some MediaRenderers may only support one type of content (e.g. audio or still images), whereas other MediaRenderers may support a wide variety of content including video, audio, still images.

The control point coordinates and manages the operation of the MediaServer and MediaRenderer as directed by the user (e.g. play, stop, pause) in order to accomplish the desired task (e.g. play "MyFavorite" music). Additionally, the control point provides the UI (if any) for the user to interact with in order to control the operation of the device(s) (e.g. to select the desired content). The layout of the control point's UI and the functionality that it exposes is implementation dependent and determined solely by the control point's manufacturer. Some examples of a control point might include a TV with a traditional remote control or a wireless PDA-like device with a small display.

Note: The above descriptions talk about devices "sending/receiving content to/from the home network." In the context of the AV Architecture, this includes point-to-point connections such as an RCA cable that is used to connect a VCR to a TV. The AV Architecture treats this type of connection as a small part (e.g. segment) of the home network. Refer to the ConnectionManager service [3] for more details.

As described above, the AV Architecture consists of three distinct components that perform well-defined roles. In some cases, these components will exist as separate, individual UPnP devices. However, this need not be the case. Device manufacturers are free to combine any of these logical entities into a single physical device. In such cases, the individual components of these combo devices may interact with each other using either the standard UPnP control protocols (e.g. SOAP over HTTP) or using some private communication mechanism. In either case, the function of each logical entity remains unchanged. However, in the later case, since the communication between the logical entities is private, the individual components will not be able to communicate with other UPnP AV devices that do not implement the private protocol.

As shown in Figure 3, the control point is the only component that initiates UPnP actions. The control point requests to configure the MediaServer and MediaRenderer so that the desired content flows from the MediaServer to the MediaRenderer (using one of the transfer protocols and data formats that are supported by both the MediaServer and MediaRenderer). The MediaServer and MediaRenderer do invoke any UPnP actions to the control point. However, if needed, the MediaServer and/or MediaRenderer may send event notifications to the control point in order to inform the control point of a change in the MediaServer's/MediaRenderer's internal state.

The MediaServer and MediaRenderer do not control each other via UPnP actions. However, in order to transfer the content, the MediaServer and MediaRenderer use an "out-of-band" (e.g. a non-UPnP) transfer protocol to directly transmit the content. The control point is not involved in the actual transfer of the content. It simply configures the MediaServer and MediaRenderer as needed and initiates the transfer of the content. Once the transfer begins, the control point "gets out of the way" and is no longer needed to complete the transfer.

However, if desired by the user, the control point is capable of controlling the flow of the content by invoking various AVTransport actions such as Stop, Pause, FF, REW, Skip, Scan, etc. Additionally, the control point is also able to control the various rendering characteristics on the Renderer device such as Brightness, Contrast, Volume, Balance, etc.

5.1 MediaServer

The MediaServer is used to locate content that is available via the home network. MediaServers include a wide variety of devices including VCRs, DVD players, satellite/cable receivers, TV tuners, smartphones, radio tuners, CD players, audio tape players, MP3 players, PCs, etc. A MediaServer's primary purpose is to allow control points to enumerate (i.e. browse or search for) content items that are available for the user to render. The MediaServer contains a ContentDirectory service [2], a ConnectionManager service [3], and an optional AVTransport service [1] (depending on the supported transfer protocols).

Some MediaServers are capable of transferring multiple content items at the same time, e.g. a hard-disk-based audio jukebox may be able to simultaneously stream multiple audio files to

the network. In order to support this type of MediaServer, the ConnectionManager assigns a unique Connection ID to each "connection" (i.e. each stream) that is made. This ConnectionID allows a third-party control points to obtain information about active connections of the MediaServer.

5.1.1 ContentDirectory Service

This service provides a set of actions that allow the control point to enumerate the content that the Server can provide to the home network. The primary action of this service is *ContentDirectory::Browse()*. This action allows control points to obtain detailed information about each Content Item that the Server can provide. This information (i.e. meta-data) includes properties such as its name, artist, date created, size, etc. Additionally, the returned meta-data identifies the transfer protocols and data formats that are supported by the Server for that particular Content Item. The control point uses this information to determine if a given MediaRenderer is capable of rendering that content in its available format.

5.1.2 ConnectionManager Service

This service is used to manage the connections associated with a particular device. The primary action of this service (within the context of a MediaServer) is *ConnectionManager::PrepareForConnection()*. When implemented, this optional action is invoked by the control point to give the Server an opportunity to prepare itself for an upcoming transfer. Depending on the specified transfer protocol and data format, this action may return the InstanceID of an AVTransport service that the control point can use to control the flow of this content (e.g. Stop, Pause, Seek, etc). As described below, this InstanceID is used to distinguish multiple (virtual) instances of the AVTransport service, each of which is associated with a particular connection to Renderer. Multiple (virtual) instances of the AVTransport service allow the MediaServer to support multiple Renderers at the same time. When the control point wants to terminate this connection, it should invoke the MediaServer's *ConnectionManager::ConnectionComplete()* action (if implemented) to release the connection.

If the *ConnectionManager::PrepareForConnection()* action is not implemented, the control point is only able to support a single Renderer at an given time. In this case, the control point should use InstanceID=0.

5.1.1 AVTransport Service

This (optional) service is used by the control point to control the "playback" of the content that is associated with the specified AVTransport. This includes the ability to Stop, Pause, Seek, etc. Depending on the supported transfer protocols and/or data formats, a MediaServer may or may not implement this service. If supported, the MediaServer can distinguish between multiple instances of the service by using the InstanceID that is included in each AVTransport action. New instances of the AVTransport service are created via the ConnectionManager's *ConnectionManager::PrepareForConnection()* action. A new Instance Id is allocated for each new service Instance.

5.2 MediaRenderer

The MediaRenderer is used to render (e.g. display and/or listen to) content obtained from the home network. This includes a wide variety of devices including TVs, stereos, speakers, hand-held audio players, music controlled water-fountain, etc. Its main feature is that it allows the control point to control how content is rendered (e.g. Brightness, Contrast, Volume, Mute, etc). Additionally, depending on the transfer protocol that is being used to obtain the content from the network, the MediaRenderer may also allow the user to control the flow of the content (e.g. Stop, Pause, Seek, etc). The MediaRenderer includes a RenderingControl service [6], a ConnectionManager service, and an optional AVTransport service (depending on which transfer protocols are supported).

In order to support rendering devices that are capable of handling multiple content items at the same time (e.g. an audio mixer such as a Karaoke device), the RenderingControl and

AVTransport services contain multiple independent (logical) instances of these services. Each (logical) instance of these services is bound to a particular incoming connection. This allows the control point to control each incoming content item independently from each other.

Multiple logical instances of these services are distinguished by a unique 'InstanceID' which references the logical instance. Each action invoked by the control point contains the Instance ID that identifies the correct instance.

5.2.1 RenderingControl Service

This service provides a set of actions that allow the control point to control how the Renderer renders a piece of incoming content. This includes rendering characteristics such as Brightness, Contrast, Volume, Mute, etc. The RenderingControl service supports multiple, dynamic instances, which allows a Renderer to "mix together" one or more content items (e.g. a Picture-in-Picture window on a TV or an audio mixer device). New instances of the service are created by the *ConnectionManager::PrepareForConnection()* action. If the *ConnectionManager::PrepareForConnection()* action is not implemented the default value 0 should be used for InstanceID.

5.2.2 ConnectionManager Service

This service is used to manage the connections associated with a device. Within the context of a MediaRenderer, the primary action of this service is the *ConnectionManager::GetProtocolInfo()* action. This action allows a control point to enumerate the transfer protocols and data formats that are supported by the MediaRenderer. This information is used to predetermine if a MediaRenderer is capable of rendering a specific content item. A MediaRenderer may also implement the optional *ConnectionManager::PrepareForConnection()* action. This action is invoked by the control point to give the Render an opportunity to prepare itself for an upcoming transfer. Additionally, this action assigns a unique ConnectionID that can be used by a 3rd-party control point to obtain information about the connections that the MediaRenderer is using. Also, depending on the specified transfer protocol and data format being used, this action may return a unique AVTransport InstanceID that the control point can use to control the flow of the content (e.g. Stop, Pause, Seek, etc). (Refer to 5.2.3 below for additional details). Lastly, the *ConnectionManager::PrepareForConnection()* action also returns a unique RenderingControl InstanceID which can be used by the control point to control the rendering characteristics of the associated content as described above. When the control point wants to terminate a connection, it should invoke the Renderer's *ConnectionManager::ConnectionComplete()* action (if implemented) to release the connection. If the *ConnectionManager::PrepareForConnection()* action is not implemented the default value 0 should be used for InstanceID.

5.2.3 AVTransport Service

This (optional) service is used by the control point to control the flow of the associated content. This includes the ability to Play, Stop, Pause, Seek, etc. Depending on transfer protocols and/or data formats that are supported, the Renderer may or may not implement this service. In order to support MediaRenderers that can simultaneously handle multiple content items, the AVTransport service may support multiple logical instances of this service. As described above, AVTransport InstanceIDs are allocated by the *ConnectionManager::PrepareForConnection()* action to distinguish between multiple service instances.

5.3 Control point

Control points coordinate the operation of the MediaServer and the MediaRenderer, usually in response to user interaction with the control point's UI. A control point is not a UPnP device, e.g. it is not visible as a device on the network, since it does not provide any UPnP services. Conversely, the control point invokes services on other UPnP devices in order to trigger some desired behavior of the remote device.

The following describes a generic control point algorithm that can be used to interact with a wide variety of MediaServer and MediaRenderer implementations.

a) **Discover AV Devices:** Using UPnP's Discovery mechanism, MediaServers and MediaRenderers in the home network are discovered.

b) **Locate Desired Content:** Using the Server's *ContentDirectory::Browse()* or *ContentDirectory::Search()* actions, a desired Content Item is located. The information returned by *ContentDirectory::Browse()*/*Search()* includes the transfer protocols and data formats that the MediaServer supports to transfer the content to the home network.

c) **Get Renderer's Supported Protocols/Formats:** Using the MediaRenderer's *ConnectionManager::GetProtocolInfo()* action a list of transfer protocols and data formats supported by the MediaRenderer is returned to the control point.

d) **Compare/Match Protocols/Formats and Check Playability:** The protocol/format information returned by the ContentDirectory for the desired Content Item is matched with the protocol/format information returned by the MediaRenderer's *ConnectionManager::GetProtocolInfo()* action. The control point selects a transfer protocol and data format that are supported by both the MediaServer and MediaRenderer. For a more detailed check of the MediaRenderer's ability to play the desired Content Item, the ControlPoint can invoke the optional *ConnectionManager::GetRendererItemInfo()* action. This action returns more detailed information about the playback capabilities of the MediaRenderer for this Content Item, such as DRM status, supported video resolution.

e) **Configure Server/Renderer:** The device's *ConnectionManager::PrepareForConnection()* action (if implemented) informs the MediaServer and MediaRenderer that an outgoing/incoming connection is about to be made using the specified transfer protocol and data format that was previously selected. Depending on the selected transfer protocol, either the MediaServer or MediaRenderer will return an AVTransport InstanceID. This InstanceID is used in conjunction with the device's AVTransport service (i.e. the device returning the AVTransport InstanceID) to control the flow of the content (e.g. *AVTransport::Play()*, *AVTransport::Stop()*, *AVTransport::Pause()*, *AVTransport::Seek()*, etc). Additionally, the Renderer will return a RenderingControl InstanceID that is used by the control point to control the Rendering characteristics of the content.

Note: Since *ConnectionManager::PrepareForConnection()* is an optional action, there may be situations in which either the MediaServer and/or MediaRenderer do not implement *ConnectionManager::PrepareForConnection()*. When this occurs and neither MediaServer nor MediaRenderer return an AVTransport InstanceID, the control point uses an InstanceID=0 to control the flow of the content. Refer to the ConnectionManager and AVTransport service [1] for details.

f) **Select Desired Content:** Using the AVTransport service (whose InstanceID is returned by either the Server or Renderer), invoke the *AVTransport ::SetAVTransportURI()* action to identify the content item that needs to be transferred.

g) **Start Content Transfer:** Using the AVTransport service, invoke one of the transport control actions as desired by the user (e.g. *AVTransport::Play()*, *AVTransport::Stop()*, *AVTransport::Seek()*, etc).

h) **Adjust Rendering Characteristics:** Using the MediaRenderer's RenderingControl service [6], invoke any rendering control actions as desired by the user (e.g. adjust brightness, contrast, volume, mute, etc). More sophisticated rendering characteristics (e.g. rotation, red-eye removal) can be set by means of the action *RenderingControl::SetTransforms()*. The available rendering characteristics which can be controlled in this manner are listed by *RenderingControl:: GetAllowedTransforms()*action.

i) **Repeat: Select Next Content:** Using either the *AVTransport::SetAVTransportURI()* or *AVTransport::SetNextAVTRansportURI()* actions, identify the next content item that is to be transferred from the same Server to the same Renderer. Repeat as needed.

j) **Cleanup Server/Renderer:** When the session is terminated and MediaServer and MediaRenderer are no longer needed in the context of the session, the MediaServer's and MediaRenderer's *ConnectionManager::ConnectionComplete()* action is invoked to close the MediaServer's connection.

Based on the interaction sequence shown above, the following diagram chronologically illustrates the typical interaction sequence between the control point and the MediaServer and MediaRenderer.

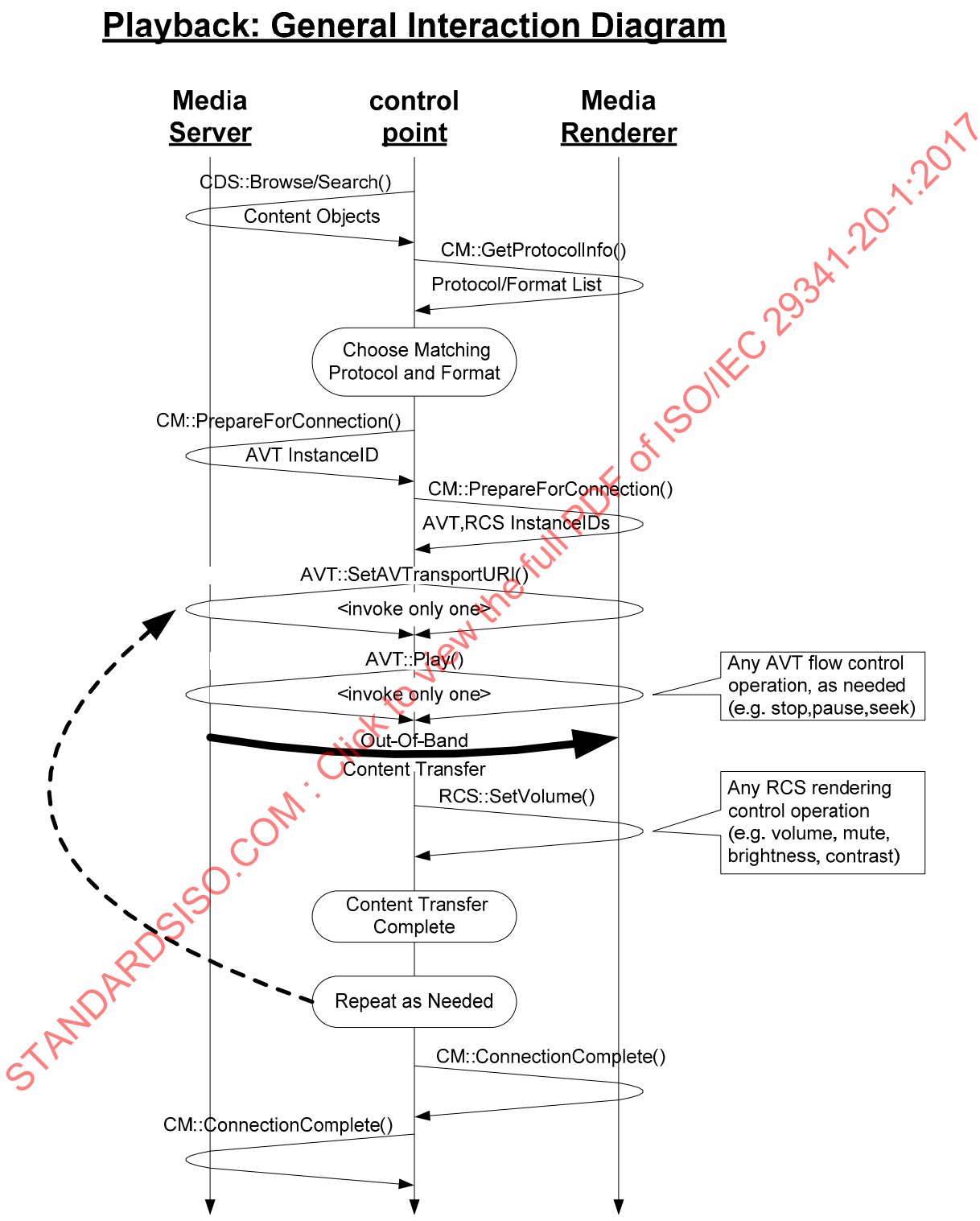

Figure 4 — General Interaction Diagram of the 3-Box model

The 3-Box model is the most comprehensive UPnP interaction model. It is also possible to combine the control point with services, to make a combo device. These scenarios are known as 2-Box models and are explained below.

5.3.1 2-Box model: Control point with Decoder

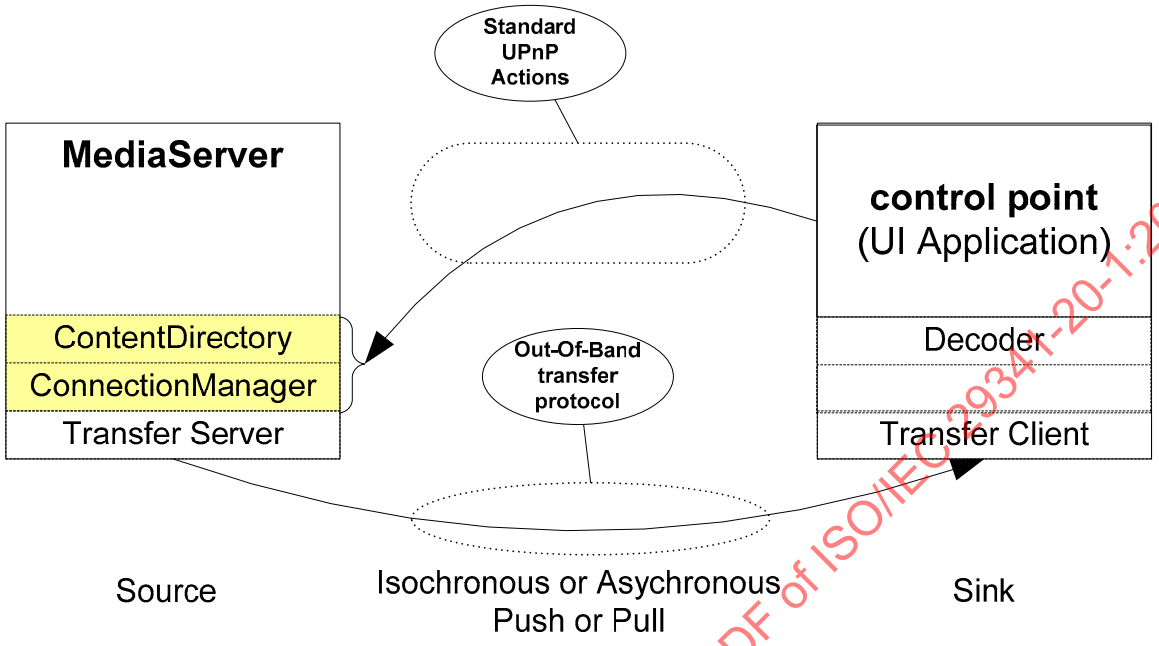

Figure 5 — Control point with Decoder

As shown in Figure 5, the content playback scenario involves two distinct UPnP components: a MediaServer, and a UPnP control point with Decoder. These two components (each with a well-defined role) work together to accomplish the task. In this scenario, the MediaServer contains (entertainment) content that the user wants to render (e.g. display or listen to) on the apparatus. The user interacts with the control point's UI to locate and select the desired content on the MediaServer and to play it back by means of its own Decoder.

The state of the system can not be tracked by any other UPnP control points, since the out of band transfer is not registered at the server or the playback device due to the absence of the AVTransport service. This scenario explains the most simplified UPnP AV interaction model.

Note that the control point in this scenario only interacts with the MediaServer services.

Note that the "Sink" in this scenario is not a MediaRenderer and not even a UPnP device.

5.3.2 2-Box model: Control point with Content

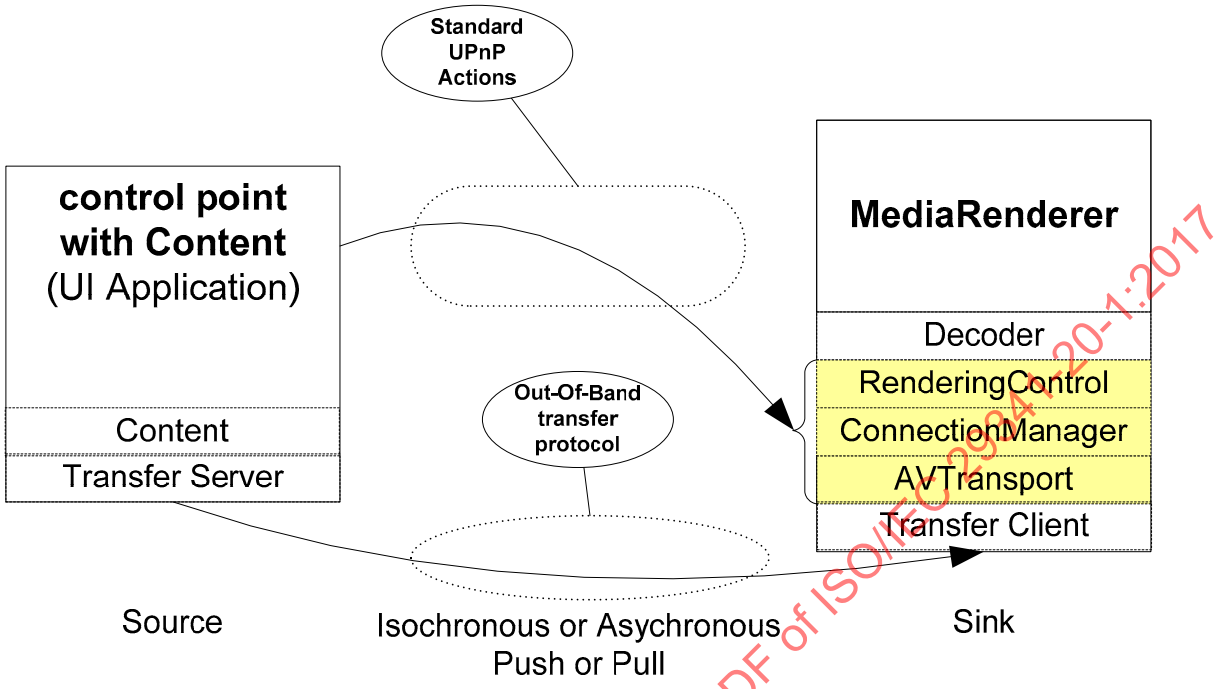

Figure 6 — Control point with Content

As shown in Figure 6, the content playback scenario involves two distinct UPnP components: a Control Point with content and a MediaRenderer. These two components (each with a well-defined role) work together to accomplish the task. In this scenario, the control point has capabilities like a normal MediaServer for serving content and contains (entertainment) content that the user wants to render (e.g. display or listen to) on the Device. The user interacts with the UI on the control point to locate and select the desired content by means of internal communication and to play it back using the MediaRenderer.

Note that the control point in this scenario only interacts with the MediaRenderer services.

5.4 Tracking streams in the network

The out of band streams are trackable by other UPnP control points on the network if:

- The optional *ConnectionManager::PrepareForConnection()* action on a ConnectionManager service is implemented (either on the Media Server or the Media Renderer side).
- Or when the playback device contains an AVTransport service.

6 Example Playback Scenarios

As described above, the AV Architecture is designed to support arbitrary transfer protocols and data formats. However, in some cases, certain devices are intentionally designed to support a single transfer protocol and/or data format only. For example, a manufacturer may want to deliver a product that targets a particular price-point and/or market segment. In these cases, some AV devices may combine one or more logical entities into a single physical device.

Subclauses 6.1–6.7 illustrate the flexibility of the generic Device Interaction Model algorithm. Each of the following interaction diagrams are variations of the generic diagram with various

steps omitted. These omitted steps are not included because the particular scenario does not require them.

6.1 3-Box model: Isochronous-Push (IEC61883/IEEE1394)

When using an isochronous transfer protocol (e.g.IEC61883/ IEEE1394), the underlying transfer mechanism provides real-time content transfer between the MediaServer and MediaRenderer. This ensures that individual packets of content are transferred within a certain (relatively small) period of time. This real-time behavior allows the MediaRenderer to provide the user with smooth-flowing rendering of the content without implementing a read-ahead buffer. In this environment, the flow of the content is controlled by the MediaServer. The MediaRenderer immediately renders the content that it receives from the MediaServer. Refer to the diagram below for details.

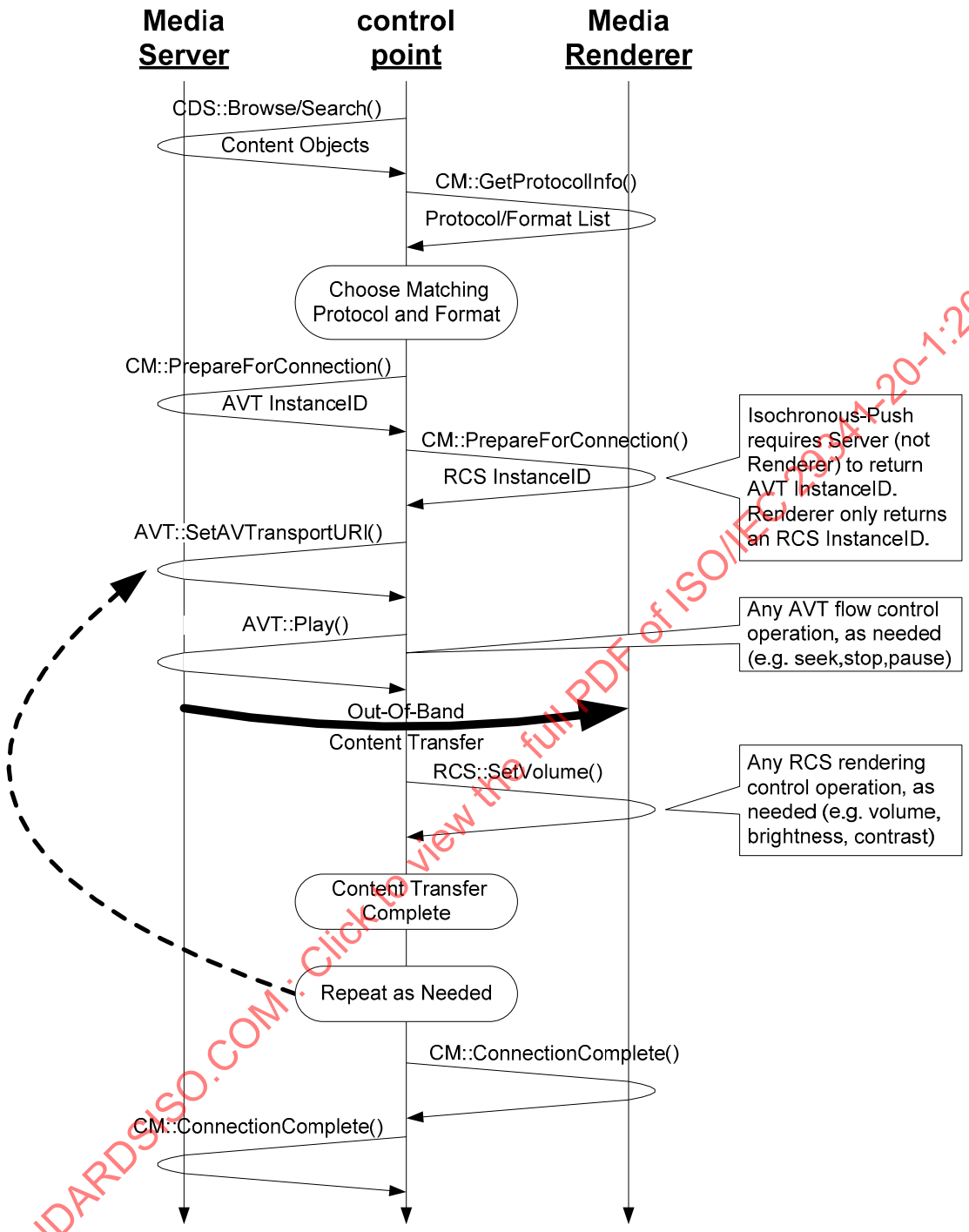

Figure 7 — 3-Box Model: Isochronous-Push transfer protocols

6.2 3-Box model: Asynchronous-Pull (e.g. HTTP GET)

In this example, the transfer protocols that are used do not provide real-time guarantees. The arrival of a particular packet of content is unpredictable relative to the previous packets. Unless corrected, this causes the content to be rendered with certain undesirable anomalies (e.g. detectable latencies, jitter, etc.). In order to compensate for these types of transfer mechanisms, a Renderer device typically implements a read-ahead storage buffer in which the Renderer reads-ahead of the current output and places the data into a buffer until the contents are needed. This allows the MediaRenderer to smooth out any rendering anomalies that might otherwise exist. Since the MediaRenderer needs to control the flow of the content, it is obligated to provide the instance of the AVTransport service that will be used.

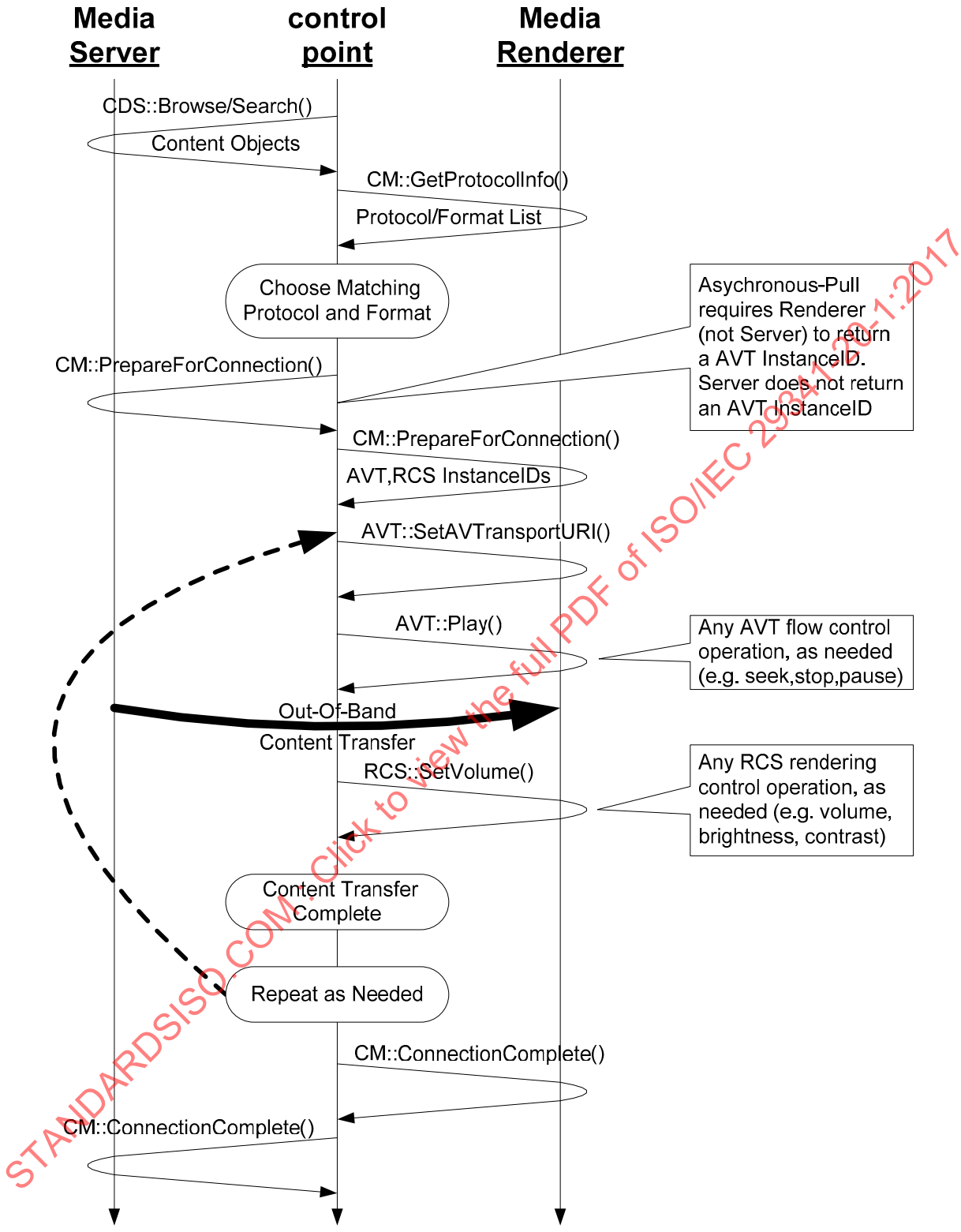

Figure 8 — 3-Box model:Asynchronus-Pull transfer protocol

6.3 2-Box model: Control point with Decoder using Isochronous-Push (e.g. IEEE-1394)

The following example illustrates how the general Device Interaction Algorithm is used to handle devices that also include integrated control point functionality (e.g. a TV), that uses the AVTranport service from the MediaServer to push the content to itself.

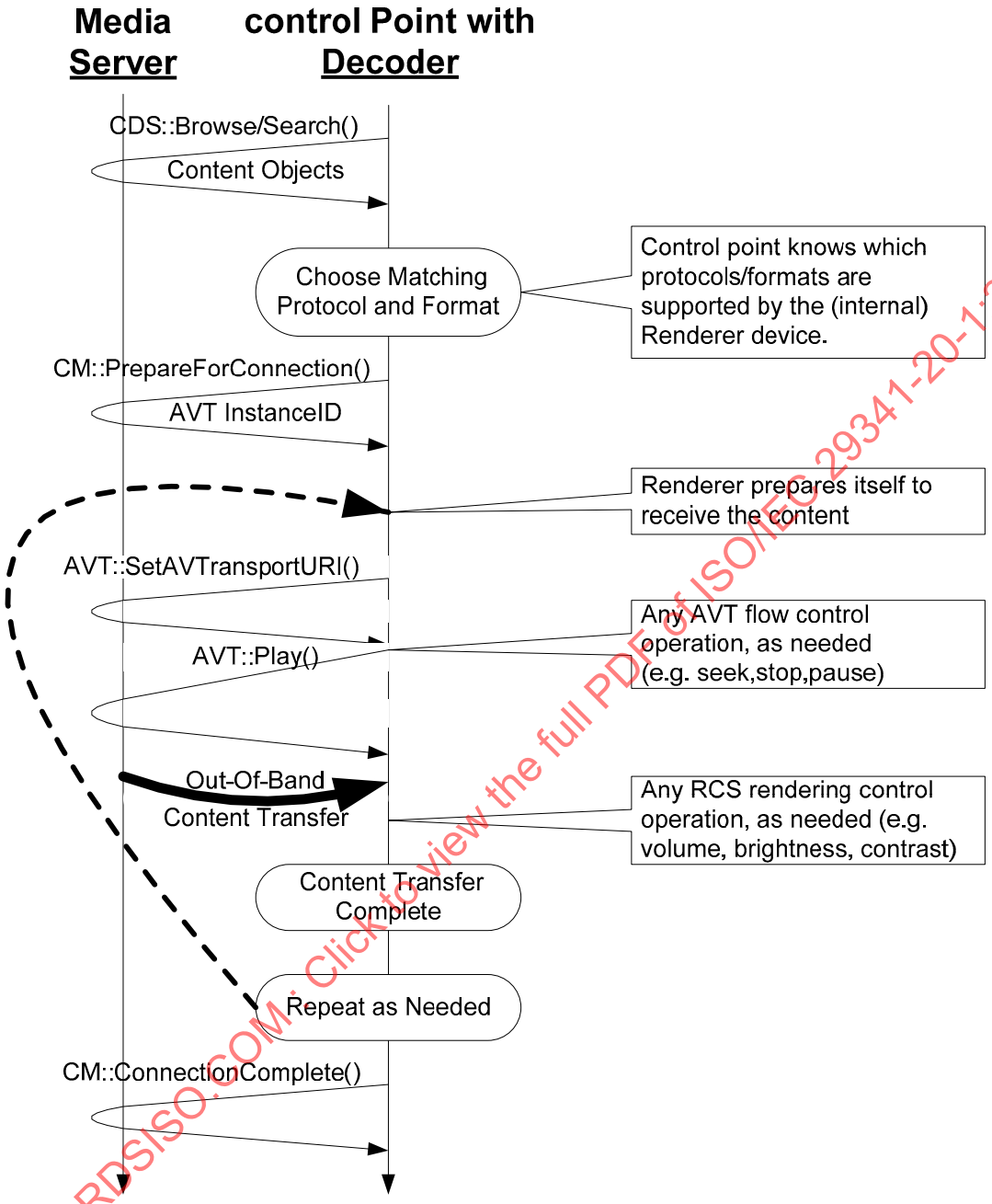

Figure 9 — 2-Box model: Control point with Decoder using Isochronous-Push

6.4 2-Box model: Control point with Decoder using Asynchronous-Pull (e.g. HTTP GET)

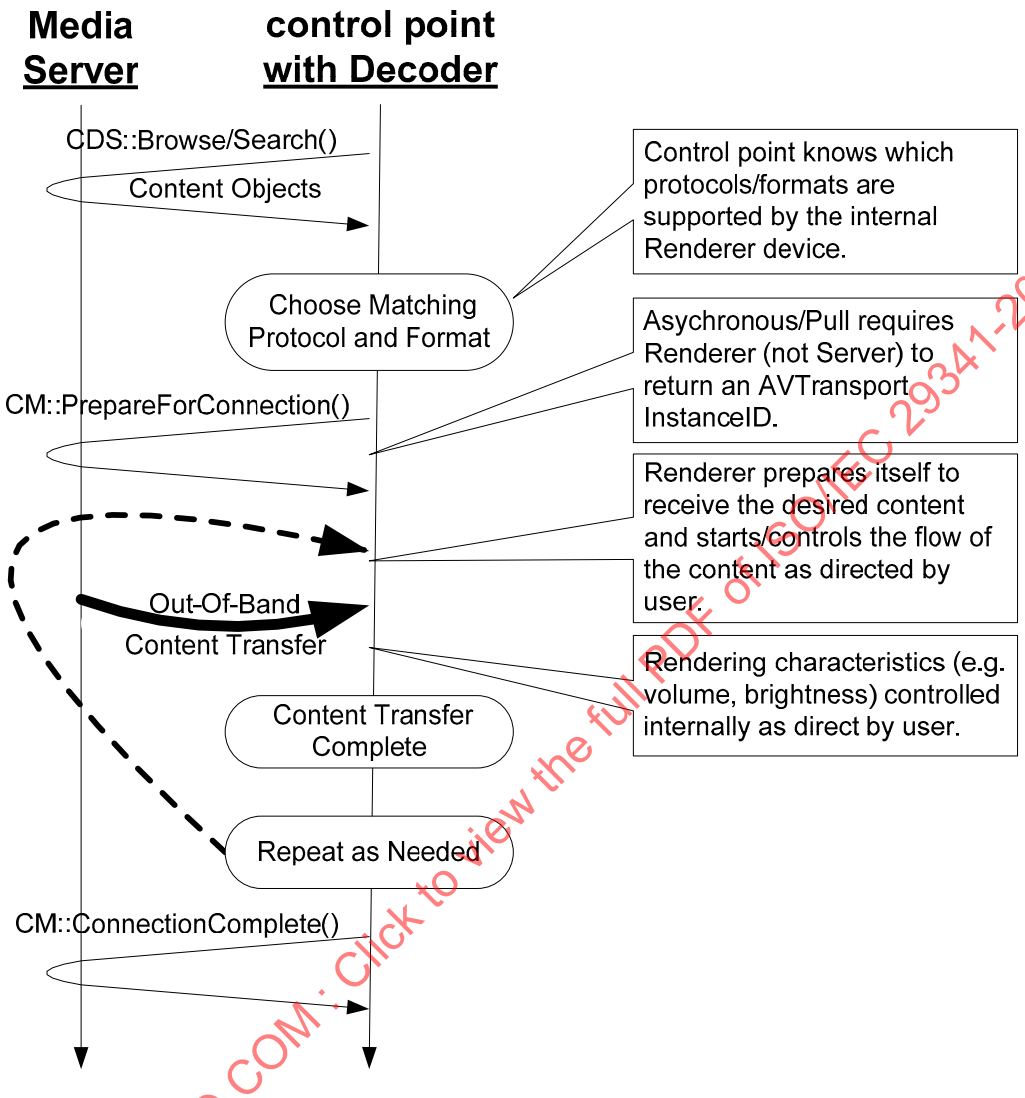

Figure 10 — 2-Box model: Control point with Decoder using Asynchronous-Pull

6.4.1 Minimal Implementation

In some cases the server only implements minimal functionality. In this case the interaction model is somewhat simpler. In this 2-Box model, the control point is being used to browse/search content on a MediaServer. This is the same as above but without the *ConnectionManager::PrepareForConnection()* and *ConnectionManager::ConnectionComplete()* actions. The actual validation of the protocol matching is done internally in the control point with Decoder. The content transfer and playback are invisible for other control points.

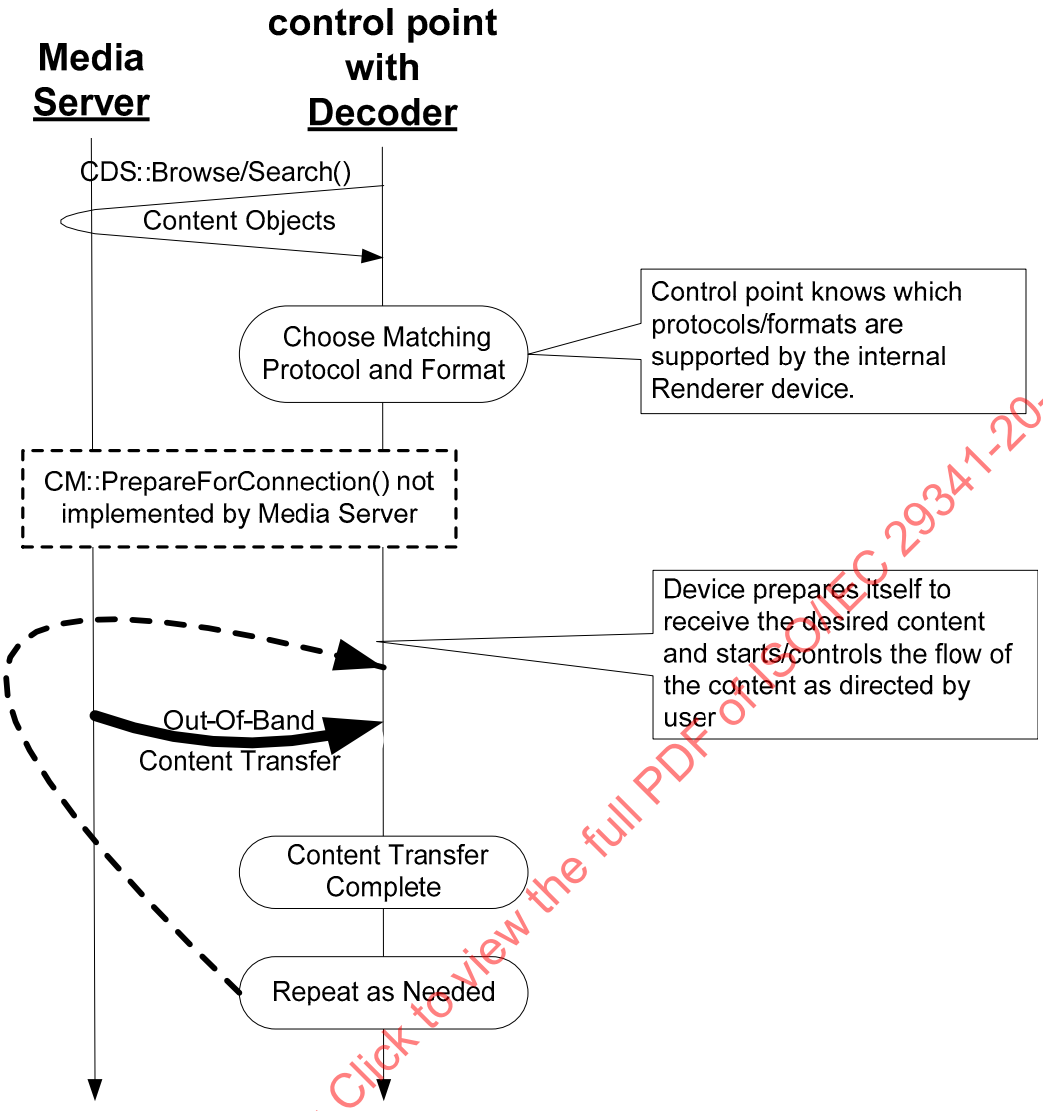

Figure 11 — 2-Box model: Minimal Implementation

6.5 2-Box model: Control point with Content using Isochronous-Push (e.g. IEEE-1394)

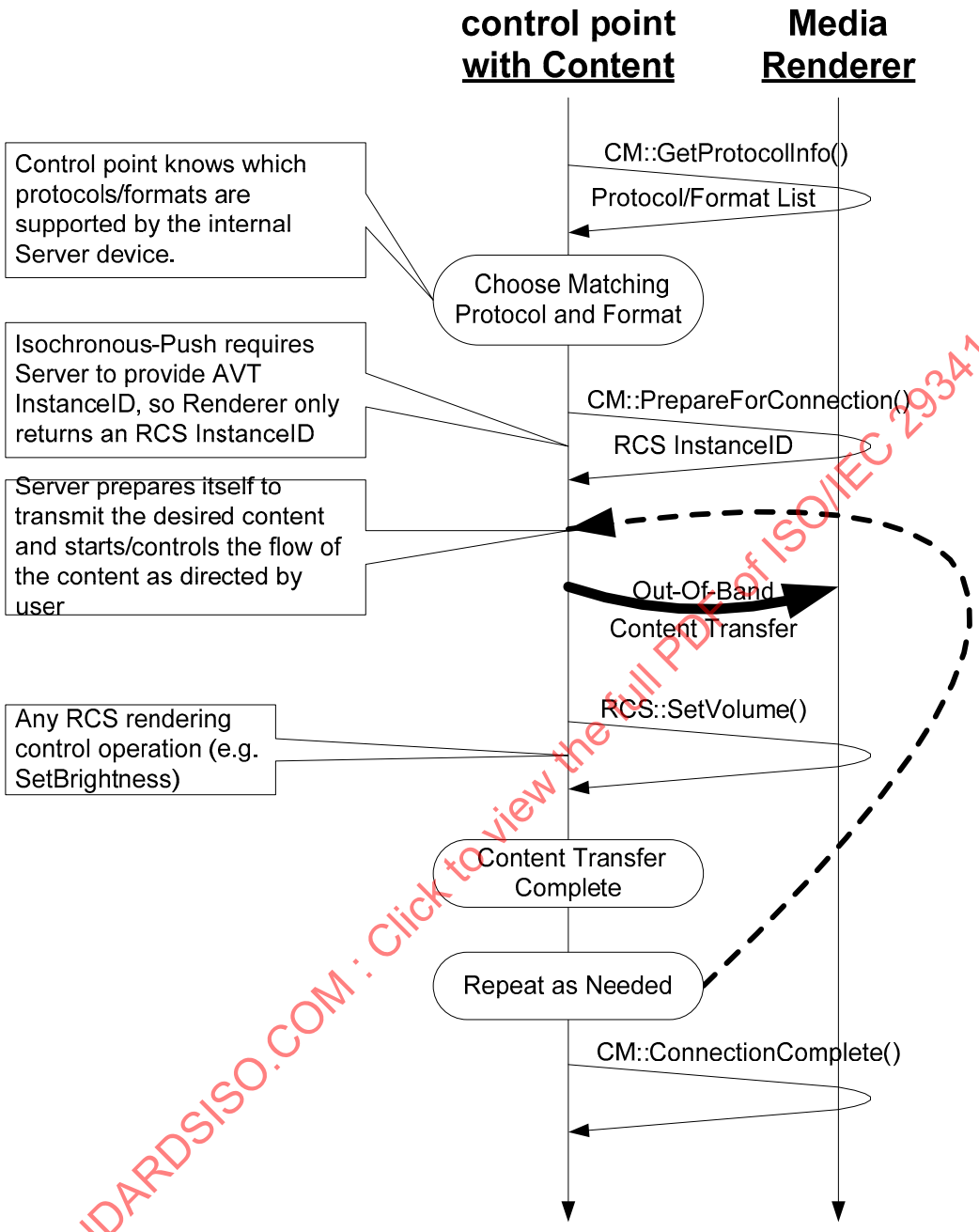

Figure 12 — 2-Box model: Control point with Content using Isochronous-Push

6.6 2-Box Model: Control point with Content using Asynchronous-Pull (e.g. HTTP GET)

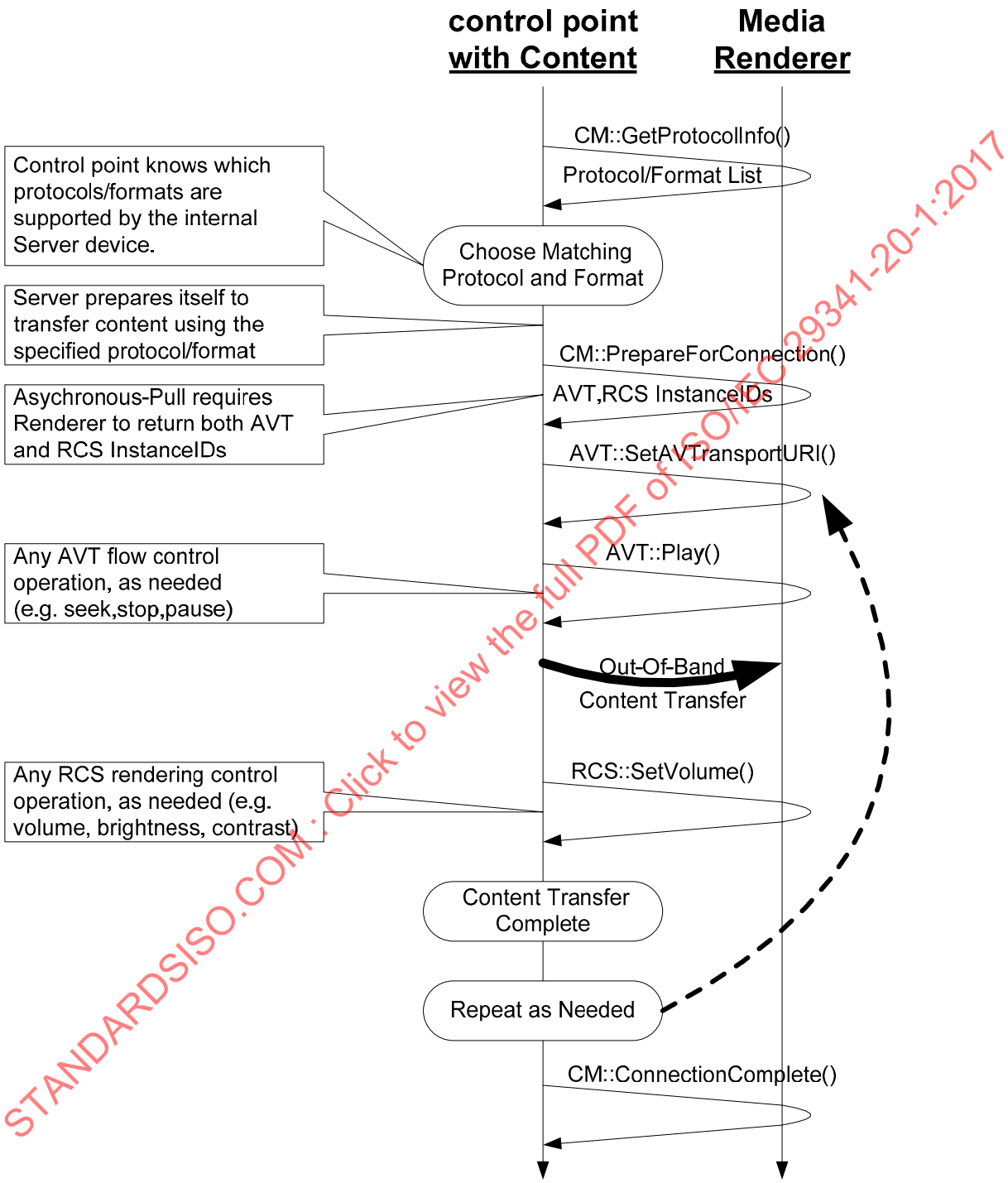

Figure 13 — 2-Box model: Control point with Content using Asynchronous-Pull

6.7 No *ConnectionManager::PrepareForConnection()* Action

In some circumstances, vendors may choose to not implement the *ConnectionManager::PrepareForConnection()* action, which (among other tasks) provides a mechanism for the control point to obtain the InstanceID of the AVTransport and RenderingControl service to use for controlling the flow and rendering characteristics of the

