RESTful Network API for Video Share

Candidate Version 1.0 – 10 Apr 2012

Open Mobile Alliance

OMA-TS-REST_NetAPI_VideoShare-V1_0-20120410-C

Contents

Figures

Tables

1. Scope

This specification defines a RESTful API for Video Share using HTTP protocol bindings.

2. References

2.1 Normative References

[Autho4API_10]	"Authorization Framework for Network APIs", Open Mobile Alliance™, OMA-ER-Autho4API-V1_0, URL: http://www.openmobilealliance.org/
[IETF_ACR_draft]	"The acr URI for anonymous users", S.Jakobsson, K.Smith, July 2011, URL: http://tools.ietf.org/html/draft-uri-acr-extension-03
[RC_API_RD]	APIs for Rich Communications Requirements, OMA-RD-RC_API-V1_0, Open Mobile Alliance, URL: http://www.openmobilealliance.org/
[REST_NetAPI_Common]	"Common definitions for RESTful Network APIs", Open Mobile Alliance™, OMA-TS-REST_NetAPI_Common-V1_0, URL:http://www.openmobilealliance.org/
[REST_NetAPI_NotificationChannel]	"RESTful Network API for Notification Channel", Open Mobile Alliance™, OMA-TS-REST_NetAPI_NotificationChannel-V1_0, URL: http://www.openmobilealliance.org/
[REST_SUP_VideoShare]	"XML schema for the RESTful Network API for Video Share", Open Mobile Alliance™, OMA-SUP-XSD-rest_netapi_VideoShare-V1.0, URL:http://www.openmobilealliance.org/
[RFC2119]	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt
[RFC2616]	"Hypertext Transfer Protocol -- HTTP/1.1", R. Fielding et. al, January 1999, URL:http://www.ietf.org/rfc/rfc2616.txt
[RFC3261]	"SIP: Session Initiation Protocol", J. Rosenberg et al., June 2002, URL: http://www.rfc-editor.org/rfc/rfc3261.txt
[RFC3966]	"The tel URI for Telephone Numbers", H. Schulzrinne, December 2004, URL: http://www.ietf.org/rfc/rfc3966.txt
[RFC3986]	"Uniform Resource Identifier (URI): Generic Syntax", R. Fielding et. al, January 2005, URL:http://www.ietf.org/rfc/rfc3986.txt
[RFC4566]	"SDP: Session Description Protocol", M. Handley et al, July 2006 , URL: http://www.ietf.org/rfc/rfc4566.txt
[RFC4627]	"The application/json Media Type for JavaScript Object Notation (JSON)", D. Crockford, July 2006, URL: http://www.ietf.org/rfc/rfc4627.txt
[SCRRULES]	"SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/
[W3C_URLENC]	HTML 4.01 Specification, Section 17.13.4 Form content types, The World Wide Web Consortium, URL: http://www.w3.org/TR/html401/interact/forms.html#h-17.13.4.1
[XMLSchema1]	W3C Recommendation, XML Schema Part 1: Structures Second Edition, URL: http://www.w3.org/TR/xmlschema-1/
[XMLSchema2]	W3C Recommendation, XML Schema Part 2: Datatypes Second Edition, URL: http://www.w3.org/TR/xmlschema-2/

2.2 Informative References

[H.263]	"RTP Payload Format for H.263 Video Streams", C. Zhu, September 1997, URL: http://www.ietf.org/rfc/rfc4627.txt
[OMADICT]	"Dictionary for OMA Specifications", Version 2.8, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_8, URL:http://www.openmobilealliance.org/
[REST_WP]	"Guidelines for RESTful Network APIs", Open Mobile Alliance™, OMA-WP-Guidelines_for_RESTful_Network_APIs, URL:http://www.openmobilealliance.org/
[RFC3550]	"RTP: A Transport Protocol for Real-Time Applications", H. Schulzrinne et al, July 2003,

URL: http://www.ietf.org/rfc/rfc3550.txt

[RFC3551] "RTP Profile for Audio and Video Conferences with Minimal Control", H. Schulzrinne et al, July 2003, URL: http://www.ietf.org/rfc/rfc3551.txt

[RFC3711] "The Secure Real-time Transport Protocol (SRTP)", M. Baugher et al, March 2004, URL: http://www.ietf.org/rfc/rfc3711.txt

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Client-side Notification URL	An HTTP URL exposed by a client, on which it is capable of receiving notifications and that can be used by the client when subscribing to notifications.
Long Polling	A variation of the traditional polling technique, where the server does not reply to a request unless a particular event, status or timeout has occurred. Once the server has sent a response, it closes the connection, and typically the client immediately sends a new request. This allows the emulation of an information push from a server to a client.
Notification Channel	A channel created on the request of the client and used to deliver notifications from a server to a client. The channel is represented as a resource and provides means for the server to post notifications and for the client to receive them via specified delivery mechanisms. For example in the case of Long Polling the channel resource is defined by a pair of URLs. One of the URLs is used by the client as a call-back URL when subscribing for notifications. The other URL is used by the client to retrieve notifications from the Notification Server.
Notification Server	A server that is capable of creating and maintaining Notification Channels.
Originator	The party that initiates a video share session.
Participant	A party that participates in a video share session, including the Originator and the Receiver.
Receiver	The party that is invited to a video share session to receive video content.
RTP/AVP	A transport protocol that denotes RTP used under the RTP Profile for Audio and Video Conferences with Minimal Control [RFC3551] running over UDP.
RTP/SAVP	A transport protocol that denotes the Secure Real-time Transport Protocol running over UDP.
Server-side Notification URL	An HTTP URL exposed by a Notification Server, that identifies a Notification Channel and that can be used by a client when subscribing to notifications.

Additionally, all definitions from the OMA Dictionary apply [OMADICT].

3.3 Abbreviations

ACR	Anonymous Customer Reference
AMR	Adaptive Multi-Rate
API	Application Programming Interface
CS	Circuit Switch
HTTP	HyperText Transfer Protocol
IANA	Internet Assigned Numbers Authority
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
OMA	Open Mobile Alliance

REST	REpresentational State Transfer
RTP	Real-time Transport Protocol
SCR	Static Conformance Requirements
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SRTP	Secure Real-time Transport Protocol
TS	Technical Specification
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	eXtensible Markup Language
XSD	XML Schema Definition

4. Introduction

The Technical Specification of the RESTful Network API for Video Share contains HTTP protocol bindings based on the requirements for video share defined in [RC_API_RD], using the REST architectural style. The specification provides resource definitions, the HTTP verbs applicable for each of these resources, and the element data structures, as well as support material including flow diagrams and examples using the various supported message body formats (i.e. XML, JSON, and application/x-www-form-urlencoding).

4.1 Version 1.0

Version 1.0 of this specification supports the following operations:

- Manage subscriptions to video share related event notifications
- Manage 1-1 video share sessions
- Notify the application about the 1-1 video share session invitation
- Notify the application about the 1-1 video share session acceptance
- Notify the application about the 1-1 video share events

In addition, this specification provides:

- Support for scope values used with authorization framework defined in [Autho4API_10]
- Support for Anonymous Customer Reference (ACR) as an end user identifier
- Support for “acr:Authorization” as a reserved keyword in a resource URL variable that identifies an end user

5. Video Share API definition

This section is organized to support a comprehensive understanding of the Video Share API design. It specifies the definition of all resources, definition of all data structures, and definitions of all operations permitted on the specified resources.

Common data types, naming conventions, fault definitions and namespaces are defined in [REST_NetAPI_Common].

The remainder of this document is structured as follows:

Section 5 starts with a diagram representing the resources hierarchy, followed by a table listing all the resources (and their URL) used by this API, along with the data structure and the supported HTTP verbs (section 5.1). What follows are the data structures (section 5.2). A sample of typical use cases is included in section 5.3, described as high level flow diagrams.

Section 6 contains the detailed specification for each of the resources. Each such subsection defines the resource, the request URL variables that are common for all HTTP commands, the possible HTTP response codes, and the supported HTTP verbs. For each supported HTTP verb, a description of the functionality is provided, along with an example of a request and an example of a response. For each unsupported HTTP verb, the returned HTTP error status is specified, as well as what should be returned in the Allow header.

All examples in section 6 use XML as the format for the message body. Application/x-www-form-urlencoded examples are provided in Appendix C, while JSON examples are provided in Appendix D. Appendix B provides the Static Conformance Requirements (SCR).

Appendix E provides the operations mapping to a pre-existing baseline specification, where applicable.

Appendix F provides a list of all lightweight resources, where applicable.

Appendix G defines authorization aspects to control access to the resources defined in this specification.

Note: Throughout this document client and application can be used interchangeably.

5.1 Resources Summary

This section summarizes all the resources used by the RESTful Network API for Video Share.

The "apiVersion" URL variable SHALL have the value "v1" to indicate that the API corresponds to this version of the specification. See [REST_NetAPI_Common] which specifies the semantics of this variable.

The figure below visualizes the resource structure defined by this specification. Note that those nodes in the resource tree which have associated HTTP methods defined in this specification are depicted by solid boxes.

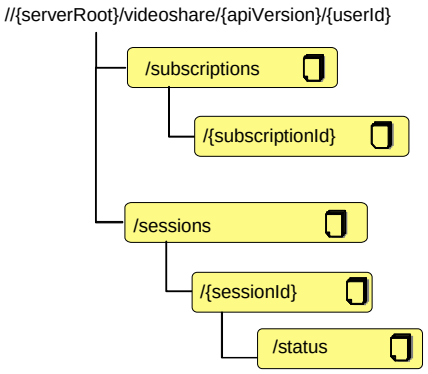

Figure 1 Resource structure defined by this specification

The following tables give a detailed overview of the resources defined in this specification, the data type of their representation and the allowed HTTP methods.

Purpose: To allow client to manage video share notifications subscriptions

Resource	URL Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions to video share event notifications	/subscriptions	VideoShareSubscriptionList (used for GET) VideoShareNotificationSubscription (used for POST) common:ResourceReference (optional alternative for POST response)	Read all active video share notification subscriptions	no	Create a new subscription for video share notifications	no
Individual subscription to video share event notifications	/subscriptions/{subsctiptionId}	VideoShareNotificationSubscription	Read an active video share notification subscription	no	no	Cancel subscription and stop corresponding notifications

Purpose: To allow client to manage 1-1 video share sessions

Resource	URL Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All 1-1 video share sessions	/sessions	VideoShareSessionInformation (used for POST) common:ResourceReference (optional alternative for POST response)	no	no	Create a new 1-1 video share session	no

Resource	URL Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Individual 1-1 video share session	/sessions/{sessionId}	VideoShareSessionInformation	Retrieve a information about a video share session	no	no	Cancel a 1-1 video share session (Originator) Decline a 1-1 video share session (Receiver) Terminate a 1-1 video share session
1-1 video share session status	/sessions/{sessionId}/status	ReceiverSessionStatus (used for POST) ReceiverSessionStatusResp (used for POST response)	no	no	Accept a video share session invitation	no

Purpose: To allow server to notify client about video share session status

Resource	URL Base URL: <Specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Client notification about 1-1 video share session invitation	Specified by client when subscription is created or provisioned	SessionInvitationNotification	no	no	Notify client about incoming 1-1 video share session invitation	no

Resource	URL Base URL: <Specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Client notification about 1-1 video share session acceptance	Specified by client when subscription is created or provisioned	SessionAcceptanceNotification	no	no	Notify client about 1-1 video share session acceptance	no
Client notification about 1-1 video share session events	Specified by client when subscription is created or provisioned	VideoShareEventNotification	no	no	Notify client about 1-1 video share session event	no
Client notification about subscription cancellation	Specified by client when subscription is created or provisioned	SubscriptionCancellationNotification	no	no	Notify client that a subscription has been cancelled (e.g. expired	no

5.2 Data Types

5.2.1 XML Namespaces

The XML namespace for the Video Share API data types is:

urn:oma:xml:rest:netapi:videoshare:1

The 'xsd' namespace prefix is used in the present document to refer to the XML Schema data types defined in XML Schema [XMLSchema1, XMLSchema2]. The 'common' namespace prefix is used in the present document to refer to the data types defined in [REST_NetAPI_Common]. The use of namespace prefixes such as 'xsd' is not semantically significant.

The XML schema for the data structures defined in the section below is given in [REST_SUP_VideoShare].

5.2.2 Structures

The subsections of this section define the data structures used in the Video Share API.

Some of the structures can be instantiated as so-called root elements.

For structures that contain elements which describe a user identifier, the statements in section 6 regarding 'tel', 'sip' and 'acr' URI schemes apply.

5.2.2.1 Type: VideoShareSessionInformation

This type represents information about a video share session.

Element	Type	Optional	Description
originatorAddress	xsd:anyURI	No	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator. When the application acts on behalf of the Originator, the originatorAddress MUST have the same value as the {userId} fragment in the resource URL path if the {userId} is also part of the request URL.
originatorName	xsd:string	Yes	Name of the Originator
receiverAddress	xsd:anyURI	Yes	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Receiver. It SHALL be present in request bodies during resource creation in case of video sharing without CS voice call. It SHALL not be present in request bodies during resource creation in case of video sharing with CS voice call. The server can get the receiverAddress using the callObjectRef received in the request bodies during resource creation in case of video sharing with CS voice call. When the application acts on behalf of the Receiver, the receiverAddress MUST have the same value as the {userId} fragment in the resource URL path if the {userId} is also part of the request URL.

callObjectRef	xsd:anyURI	Yes	Reference to the CS voice object (to which the Receiver is linked). It SHALL be present in request bodies during resource creation in case of video sharing with CS voice call. It SHALL not be present in request bodies during resource creation in case of video sharing without CS voice call.
receiverName	xsd:string	Yes	Name of the Receiver. MAY be present when receiverAddress is present.
mediaURL	xsd:anyURI	Yes	The video media URL from where the video content can be retrieved. It SHALL not be present in the POST request during resource creation if liveVideo is set to true. If it is present in the POST operation during resource creation, the server could fetch the video content using this URL. If it is not present in the POST request during resource creation and the liveVideo is not set to true (not live video), the video content is included in the HTTP body. The HTTP body can be represented as multipart/form-data entity bodies, as specified in 5.2.5.
size	xsd: unsignedLong	Yes	The size of the video file in octets
mediaInformation	MediaInformation [0..unbounded]	Yes	List of the information about the video content such as codecs, transport protocols. It SHALL be present if the video is live (liveVideo is set to true). It SHALL not be present if the video is not live (the liveVideo is not set to true).
liveVideo	xsd:boolean	Yes	Indicates whether the video is live video (true) or video clip (false). This element MUST be present and set to “true” if the video is live. Default value is ‘false.’
status	SessionStatus	Yes	Connection status of the 1-1 video share session. Set by the server (see Table 1 for detailed information).. SHALL NOT be present in request bodies during session resource creation by the application of the Originator.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a

			resource on the server. This element SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
resourceURL	xsd:anyURI	Yes	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named videoShareSessionInformation of type VideoShareSessionInformation is allowed in request and/or response bodies.

Note that the clientCorrelator is used for purposes of error recovery as specified in [REST_NetAPI_Common], and internal client purposes. The server is NOT REQUIRED to use the clientCorrelator value in any form in the creation of the URL of the resource. The specification [REST_NetAPI_Common] provides a recommendation regarding the generation of the value of this field.

The following table gives detailed information about when and what "status" value is set:

Value of "status"	Set in the following occasions:
Initial	Set by server when receiving session resource creation request or when receiving session invitation notification
Connected	Set by server when "status" is set to "Connected" in "receiverSessionStatus " root element when application of Receiver accepts a video share session invitation or Set by server when "status" is set to "Connected" in "receiverSessionStatus" of "sessionAcceptanceNotification" root element when receiving session acceptance notification.
Terminated	Set by server when receiving delete session request or when receiving video share event notifications with eventType of "SessionEnded"/"SessionCancelled"/"Declined"

Table 1: 1-1 Video share session status

5.2.2.2 Type: SessionInvitationNotification

This type represents a video share session invitation notification.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to video share notifications. See [REST_ NetAPI_Common].

link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification. The server MUST include links defined for SessionInvitationNotification (see section 6.6 for detailed information). Further, the server SHOULD include a link to the related subscription.
originatorAddress	xsd:anyURI	No	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator. When the application acts on behalf of the Originator, the originatorAddress MUST have the same value as the {userId} fragment in the resource URL path if the {userId} is also part of the request URL.
originatorName	xsd:string	Yes	Name of the Originator
receiverAddress	xsd:anyURI	Yes	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Receiver. It SHALL be present in the POST request in case of video sharing without CS voice call. It SHALL not be present in the POST request in case of video sharing with CS voice call. When the application acts on behalf of the Receiver, the receiverAddress MUST have the same value as the {userId} fragment in the resource URL path if the {userId} is also part of the request URL.
callObjectRef	xsd:anyURI	Yes	Reference to the CS voice object (to which the Receiver is linked) in case of video sharing with CS voice call. It SHALL be present in the POST request in case of video sharing with CS voice call. The server can get the callObjectRef using receiverAddress in case of video sharing with CS voice call. It SHALL not be present in the POST request in case of video sharing without CS voice call.
receiverName	xsd:string	Yes	Name of the Receiver. MAY be present when receiverAddress is present.
mediaInformation	MediaInformation [1..unbounded]	No	List of the information about the video content such as codecs, transport protocols from which the application of the Receiver can choose the ones accepted by it.

A root element named sessionInvitationNotification of type SessionInvitationNotification is allowed in notification request bodies.

5.2.2.3 Type: ReceiverSessionStatus

This type represents the status of a Receiver in the video share session.

Element	Type	Optional	Description
status	SessionStatus	No	Status of the 1-1 video share session. To indicate that the Receiver accepts the session invitation, this element MUST be set to “Connected”
mediaInformation	MediaInformation [0..2]	Yes	It contains the accepted media formats which are a sub set of the media information received in the SessionInvitationNotification. Maximum one entry for video media type and one entry for audio media type are allowed. It SHALL not be present If none of the media formats in the SessionInvitationNotification is accepted.

A root element named receiverSessionStatus of type ReceiverSessionStatus is allowed in request bodies.

5.2.2.4 Type: ReceiverSessionStatusResp

This type represents the response to the receiver session status request.

Element	Type	Optional	Description
mediaURL	xsd:anyURI	No	The media URL from where the video content can be retrieved
parameters	xsd:string [0..unbounded]	Yes	Sets of parameters for accessing the video content

A root element named receiverSessionStatusResp of type ReceiverSessionStatusResp is allowed in the response bodies.

5.2.2.5 Type: MediaInformation

This type represents the information about the video content, including media type, transport protocol, media formats and other attributes of the media type.

Element	Type	Optional	Description
mediaType	MediaType	No	Type of the media (e.g. “Audio” or “Video”) , see description of “m=” field in [RFC4566].
transportProtocol	xsd:string	No	The transport protocol of the media (e.g. “RTP/AVP”, “RTP/SAVP”, ”udp”), see description of “m=” field in [RFC4566].
mediaFormats	MediaFormat [1…unbounded]	No	List of media formats of the specified media type
bandwidth	xsd: unsignedLong	Yes	Band width in kilobits per second. See description of “b=” field in [RFC4566]
bwType	xsd:string	Yes	Currently only “CT" (Conference Total) and "AS" (Application Specific) are supported. See description of “b=” field in [RFC4566].

frameRate	xsd: decimal	Yes	The maximum video frame rate in frames/sec, defined only for video media. See description of “a=framerate” field in [RFC4566].
pTime	xsd: unsignedInt	Yes	The length of time in milliseconds represented by the media in a packet. It is probably only meaningful for audio data, but may be used with other media types if it makes sense. See description of “a=ptime” field in [RFC4566].
maxPTime	xsd: unsignedInt	Yes	This gives the maximum amount of media that can be encapsulated in each packet, expressed as time in milliseconds. For frame-based codecs, the time SHOULD be an integer multiple of the frame size. It is probably only meaningful for audio data, but may be used with other media types if it makes sense. See description of “a=maxptime” field in [RFC4566].
attributeList	Attribute [0…unbounded]	Yes	Any other attributes applicable for the specified media type

5.2.2.6 Type: Attribute

This type represents individual attribute of a list

Element	Type	Optional	Description
attributeName	xsd:string	No	Name of the attribute
attributeValue	xsd:string	Choice	The value of the attribute
<any element>	< type is defined in a schema implementing the element>	Choice	The value of the attribute. Note that element ‘any element’ can be any element from any other namespace (schema) than the target namespace. Type of such element is defined by the schema implementing the element. In XML implementations, element “any” must be qualified with the namespace prefix.

XSD modelling uses a “choice” to select either a value or <any element>.

5.2.2.7 Type: MediaFormat

This type represents information about the media formats.

Element	Type	Optional	Description
fmt	xsd:string	Yes	When “transportProtocol” is "RTP/AVP" or "RTP/SAVP”, it contains static or dynamic RTP payload type number. When static RTP payload type number is used, it

			SHALL be present in the request bodies during the session resource creation and the rest of the elements in this structure are not needed. The dynamic RTP payload type number is generated either by the server or by the client. If generated by the client, it SHALL be present in the request bodies during the session resource creation; if generated by the server, it SHALL not be present in the request bodies during the session resource creation. When dynamic RTP payload type number is used, at least “encodingName” SHALL be present. See description of “m=” field in [RFC4566].
encodingName	xsd:string	Yes	The encoding name of the RTP payload type. See description of “a=rtpmap” field in [RFC4566]. It SHALL use media subtype (e.g., “AMR” for audio, “H263-2000” for video.) when “RTP/AVP" or "RTP/SAVP" transport protocol is used, see IANA for registered media subtypes for audio and video. It SHALL be present when dynamic RTP payload type is used.
clockRate	xsd:unsignedLong	Yes	Number of samples per second, see description of “a=rtpmap” field in [RFC4566].
encodingParameter	xsd:string [0…unbounded]	Yes	List of encoding parameters for the RTP payload type, see description of “a=rtpmap” field in [RFC4566].
fmtp	xsd:string	Yes	Format specific parameters of “a=fmtp” as defined in [RFC4566]

5.2.2.8 Type: VideoShareEventNotification

This type represents a video share event notification.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The ‘callbackData’ element if it was passed by the application in the ‘callbackReference’ element when creating a subscription to video share notifications. See [REST_NetAPI_Common].
eventType	EventType	No	Type of event
eventDescription	xsd:string	Yes	Textual description of the event
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification. Depending on the value of eventType, the server MUST include links defined for

			VideoShareEventNotification (see section 6.8 for detailed information). Further, the server SHOULD include a link to the related subscription.

A root element named videoSharetEventNotification of type VideoShareEventNotification is allowed in notification request bodies.

5.2.2.9 Type: SessionAcceptanceNotification

This type represents a video session acceptance notification.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to video share notifications. See [REST_NetAPI_Common].
receiverAddress	xsd:anyURI	Yes	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Receiver. It SHALL be present in notification request bodies in case of video sharing without CS voice call. IT MAY be present in case of video sharing with CS voice call.
receiverName	xsd:string	Yes	Name of the Receiver. It MAY be present when receiverAddress is present.
callObjectRef	xsd:anyURI	Yes	Reference to the CS voice object (to which the Receiver is linked). It SHOULD be present in notification request bodies in case of video sharing with CS voice call. It SHALL not be present in request bodies during resource creation in case of video sharing without CS voice call.
status	SessionStatus	Yes	The status of the video share session. To indicate that the Receiver accepts the session invitation, this element MUST be set to "Connected".
mediaURL	xsd:anyURI	Yes	The media URL to where the live video can be delivered. It SHALL be present if the mediaInformation is not empty.
parameters	xsd:string	Yes	Sets of parameters for accessing the video

	[0..unbounded]		content. It MAY be present if the mediaURL is present.
mediaInformation	MediaInformation [0..2]	Yes	It contains the accepted media formats which are a sub set of the media information received in the VideoShareSessionInformation during resource creation in case of video sharing with live video. It SHALL not be present If none of the media formats in the POST operation during resource creation is accepted.
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification. The server MUST include links defined for SessionAcceptanceNotification (see section 6.7 for detailed information). Further, the server SHOULD include a link to the related subscription.

A root element named sessionAcceptanceNotification of type SessionAcceptanceNotification is allowed in notification request bodies.

5.2.2.10 Type: VideoShareSubscriptionList

This type represents a list of video share notification subscriptions.

Element	Type	Optional	Description
videoShareNotificationSubscription	VideoShareNotificationSubscription [0..unbounded]	Yes	Array of video share event subscriptions
resourceURL	xsd:anyURI	No	Self referring URL

A root element named videoShareSubscriptionList of type VideoShareSubscriptionList is allowed in response bodies.

5.2.2.11 Type: VideoShareNotificationSubscription

This type represents a subscription to video share notifications, i.e. VideoShareEventNotification, InvitationAcceptanceNotification and SessionInvitationNotification targeted at a particular user.

Element	Type	Optional	Description
callbackReference	common:CallbackReference	No	Client's Notification URL and OPTIONAL callbackData
duration	xsd:int	Yes	Period of time (in seconds) notifications are provided for. If set to “0” (zero), a default duration time, which is specified by the service policy, will be used. If the parameter is omitted, the notifications will continue until the maximum duration time, which is specified by the service policy, unless the notifications are stopped by deletion of subscription for notifications. This element MAY be given by the client during resource creation in order to signal the desired lifetime

			of the subscription. The server SHOULD return in this element the period of time for which the subscription will still be valid.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
resourceURL	xsd:anyURI	Yes	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named videoShareNotificationSubscription of type VideoShareNotificationSubscription is allowed in request and/or response bodies.

Regarding the clientCorrelator field, the note in section 5.2.2.1 applies.

5.2.2.12 Type: SubscriptionCancellationNotification

This type represents a subscription cancellation notification.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	CallbackData if passed by the application in the receiptRequest element during the associated subscription operation. See [REST_NetAPI_Common] for details.
reason	common:ServiceError	Yes	Reason notification is being discontinued. SHOULD be present if the reason is different from a regular expiry of the subscription.
link	common:Link[1..unbounded]	No	Link to other resources that are in relationship with the resource. There MUST be a link to the subscription that is cancelled. See section 6.9 for detailed information.

A root element named subscriptionCancellationNotification of type SubscriptionCancellationNotification is allowed in notification request bodies.

5.2.3 Enumerations

The subsections of this section define the enumerations used in the Video Share API.

5.2.3.1 Enumeration: EventType

This enumeration is used in notifications to describe the type of event which the notification is about.

Enumeration	Description
SessionCancelled	The Originator has cancelled the video share session during the invite phase.
SessionEnded	The video share session has ended.
Declined	The video share Receiver has declined the video share session invitation.
Failed	The video share has failed.

5.2.3.2 Enumeration: SessionStatus

This enumeration defines the possible values to describe the status of a video share session.

Enumeration	Description
Initial	The video share session is in initial status, the Receiver is being invited to a video share session.
Connected	The video share session is in connected status.
Terminated	The video share session is terminated.

5.2.3.3 Enumeration: MediaType

This enumeration defines the possible value for the media type.

Enumeration	Description
Video	Video type media
Audio	Audio type media

5.2.4 Values of the Link “rel” attribute

The “rel” attribute of the Link element is a free string set by the server implementation, to indicate a relationship between the current resource and an external resource. The following are possible strings (list is non-exhaustive, and can be extended):

- VideoShareSessionInformation
- ReceiverSessionStatus
- VideoShareNotificationSubscription

These values indicate the kind of resource that the link points to.

5.2.5 MIME multipart representation

In Video Share API, the session creation operation can contain actual video file content in the HTTP requests. To represent such MIME multipart messages, “multipart/form-data” format is used, where the first entry of the form is the root fields and the second entry of the form is multimedia content. Details about the structure of such messages are defined in [REST_NetAPI_Common].

5.3 Sequence Diagrams

The following sub-sections describe the resources, methods and steps involved in typical scenarios.

A sequence diagram that contains a step that involves delivering a notification, the delivery can be via the HTTP POST method or via an operation labeled “NOTIFY”. The term “NOTIFY” refers to the use of the Notification Channel [REST_NetAPI_NotificationChannel].

Note that some of the following scenarios involve both the application of the Originator and the application of the Receiver. Depending on the implementations, if the scenarios involving only the application of a Participant, the following scenarios of the application of that particular Participant apply.

5.3.1 Subscription to video share notifications

This figure below shows a scenario for an application subscribing to and unsubscribing from video share notifications.

The notification URL passed by the client during the subscription step can be a Client-side Notification URL, or a Server-side Notification URL. Refer to [REST_NetAPI_NotificationChannel] for sequence flows illustrating the creation of a Notification Channel and obtaining a Server-side Notification URL on the server-side, and its use by the client via Long Polling.

The resources:

- To subscribe to video share notifications, create a new resource under **http://{serverRoot}/videoshare/{apiVersion}/{userId}/subscriptions**
- To cancel subscription to video share notifications delete the resource under **http://{serverRoot}/videoshare/{apiVersion}/{userId}/subscriptions/{subscriptionId}**

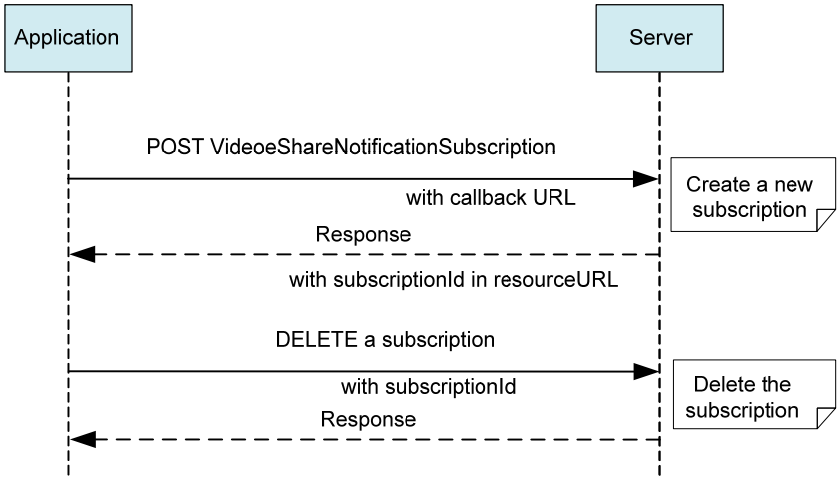

Figure 2 Subscribing to and unsubscribing from video share notifications

Outline of the flows:

1. An application subscribes to video share notifications using the POST method to submit the VideoShareNotificationSubscription data structure to the resource containing all subscriptions and receives the result resource URL containing the subscriptionId.
2. The application stops receiving notifications using DELETE with the resource URL containing the subscriptionId.

5.3.2 Normal flow of 1-1 video share session

The figure below shows a scenario for a video share session with successful result.

The resources:

- To start a 1-1 video share session, create a new resource with the VideoShareSessionInformation data structure under **http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions**
- To accept a 1-1 video share session invitation update the resource **http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}/status**
- To end a 1-1 video share session delete the resource **http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}**

5.3.2.1 1-1 video share session with recorded or stored video file

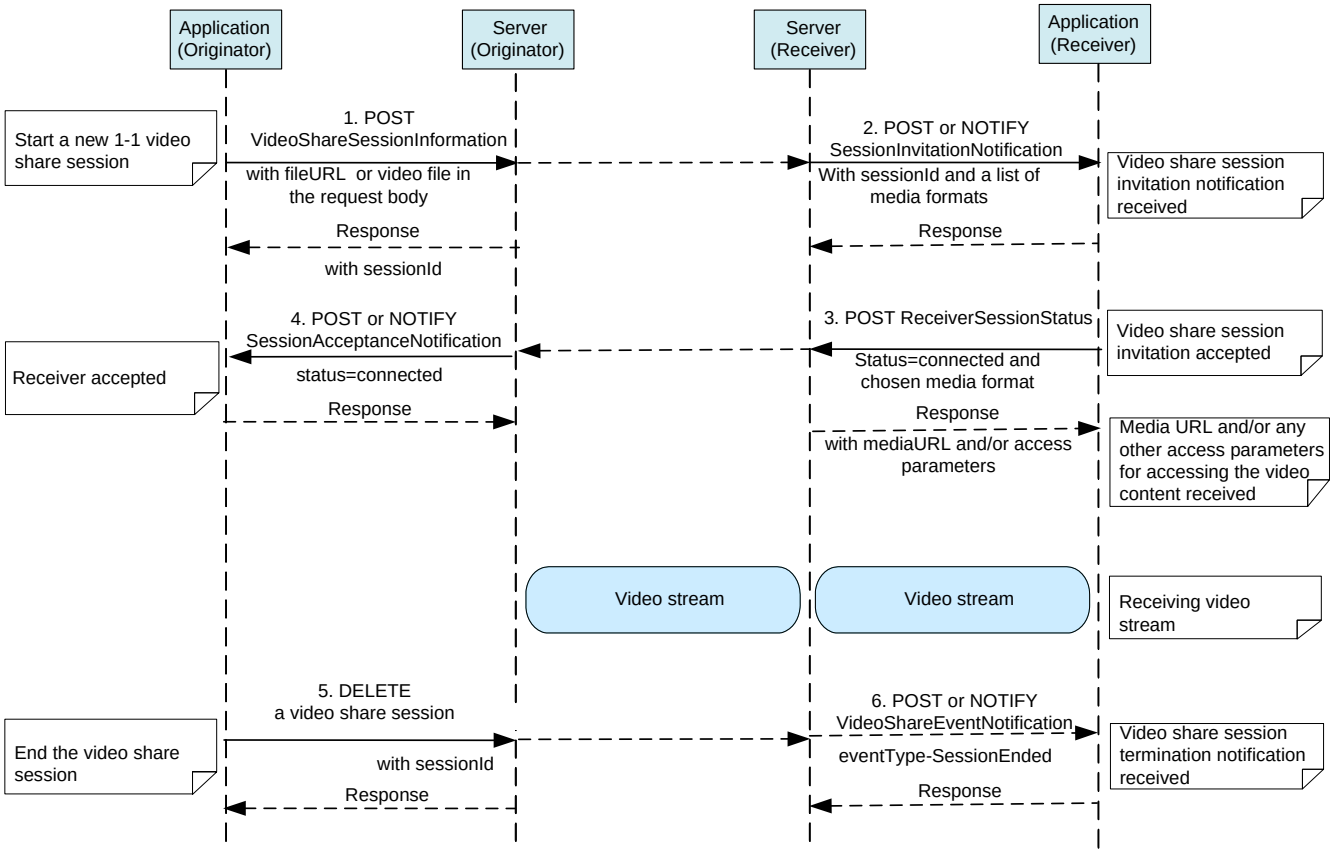

Figure 3 1-1 video share session with recorded or stored video file

Outline of the flows:

1. An application of the Originator starts a 1-1 video share session using the POST method to submit the VideoShareSessionInformation data structure containing either the recorded/stored video file or the media URL of that video file to the resource containing all video share sessions. Thereby the creation of a new video share session resource is triggered and the application of the Originator receives the resulting resource URL containing the sessionId.
2. An application of the Receiver receives a video share session invitation notification with sessionId and a list of media formats in which the media can be made available.
3. The application of the Receiver accepts the video share session invitation using the POST method to submit the ReceiverSessionStatus data structure with status set to "Connected" and the chosen media format to the resource containing the session status and receives a response containing the media URL and/or any other access parameters for accessing the media.
4. The application of the Originator receives a notification with a SessionAcceptanceNotification data structure indicating that the Receiver has accepted the invitation. The server of the Originator can start streaming the video and the application of the Receiver receives the video streaming using the media URL and access parameters received in step 3.

Note: How the application of the Receiver gets the video streaming using the received media URL and/or access parameters is out of scope.

5. After the video share session is in connected status, the application of the Originator can end the video share session at any time by using DELETE method on the resource URL of the session with sessionId
6. The application of the Receiver receives a VideoShareEventNotification data structure indicating that the session has been ended.

Note regarding steps 5 and 6: Either the application of the Originator or the application of the Receiver can end the video share session after video share session invitation has been accepted. In that case, the application on the other side of the video share session receives a VideoShareEventNotification data structure indicating that the session has been ended.

5.3.2.2 1-1 video share session with live video

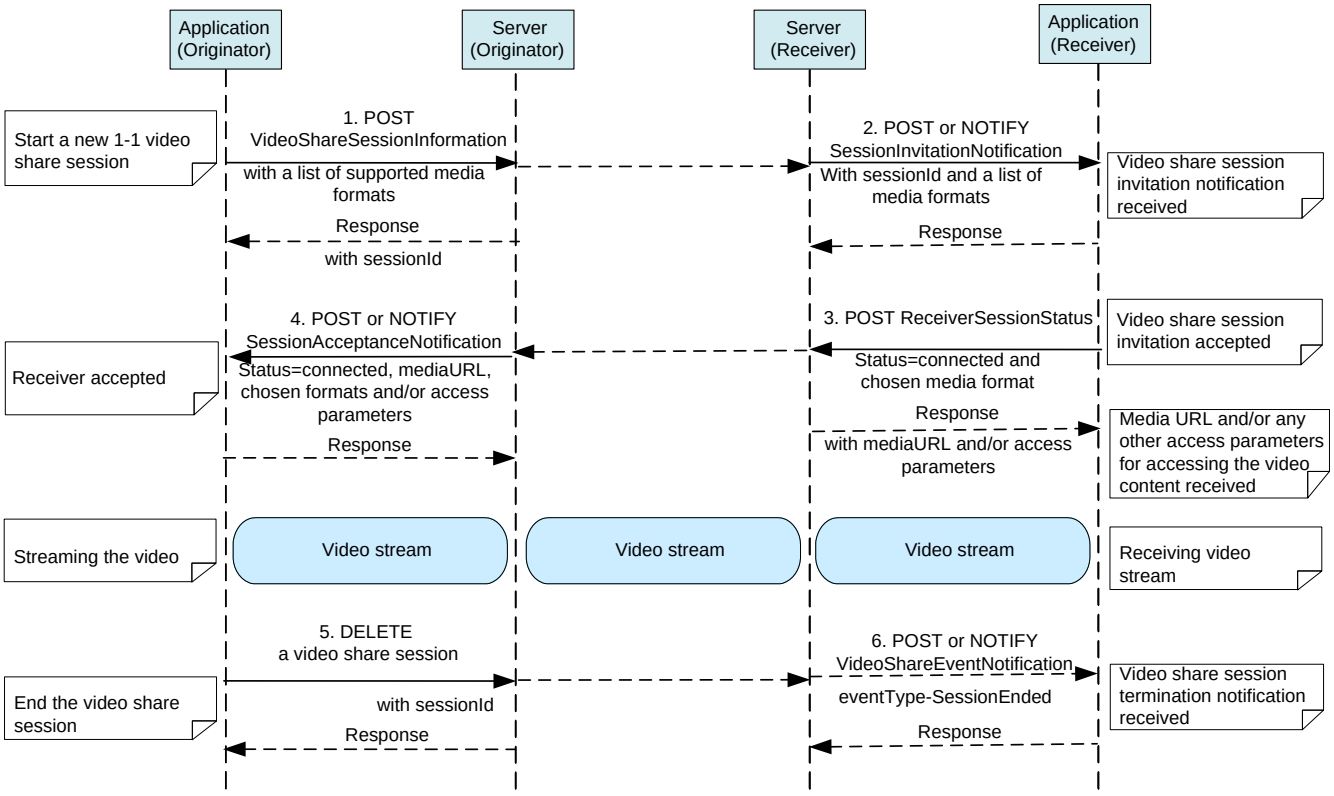

Figure 4 1-1 video share session with live video

Outline of the flows:

1. An application of the Originator starts a 1-1 video share session using the POST method to submit the VideoShareSessionInformation data structure containing a list of the media formats the application can support to the resource containing all video share sessions. Thereby the creation of a new video share session resource is triggered and the application of the Originator receives the resulting resource URL containing the sessionId.
2. An application of the Receiver receives a video share session invitation notification with sessionId and a list of media formats in which the media can be made available.
   Note: that list doesn't have to be the same as the list in step 1.
3. The application of the Receiver accepts the video share session invitation using the POST method to submit the ReceiverSessionStatus data structure with status set to "Connected" and the chosen media format to the resource containing the session status and receives a response containing the media URL and/or any other access parameters for accessing the media.
4. The application of the Originator receives a notification with a SessionAcceptanceNotification data structure indicating that the Receiver has accepted the invitation and additionally the chosen media format, the media URL and/or any other access parameters, to which the application shall subsequently send the media. The application of the Originator can start streaming the video.
5. After the video share invitation has been accepted, the application of the Originator can end the video share session at any time by using DELETE method on the resource URL of the session with sessionId
6. The application of the Receiver receives a VideoShareEventNotification data structure indicating that the session has been ended.

Note regarding steps 5 and 6: Either the application of the Originator or the application of the Receiver can end the video share session after video share session invitation has been accepted. In that case, the application of the other side of the video share session receives a VideoShareEventNotification data structure indicating that the session has been ended.

5.3.3 1-1 video share session failure

There are different causes which may lead to video share session failure, following are some options (not exclusive list):

a. The application of the Originator cancels the video share session.

b. The application of the Receiver rejects or declines the video share session invitation.

c. The video share session fails due to errors

5.3.3.1 Cancelling a 1-1 video share invitation

The figure below shows a scenario for an application of the Originator to cancel a video share session invitation.

The resources:

- To cancel a 1-1 video share session invitation delete the session resource
  http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}

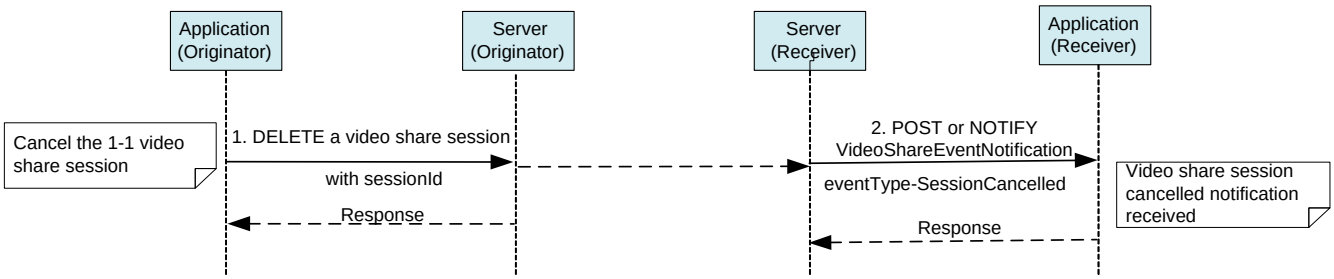

Figure 5 Cancelling a 1-1 video share invitation

Outline of the flows:

An application of the Originator has created a video share session resource triggering a video share invitation sent to the Receiver (Refer to step 1 and step 2 in 5.3.2). Subsequently:

1. The application of the Originator can cancel a 1-1 video share session invitation using the DELETE method on the resource URL of the session with sessionId and receives a response weather the request was successfully initiated.
2. The application of the Receiver receives a notification containing the VideoShareEventNotification data structure indicating that the video share session has been cancelled.

Note that cancelling a session only works before the Receiver has accepted the video share invitation. After that, the DELETE method leads to an existing session to be terminated.

5.3.3.2 Declining a 1-1 video share session invitation

The figure below shows a scenario for an application to decline a video share session invitation.

The resources:

- To decline a video share session invitation delete the session resource **http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}**

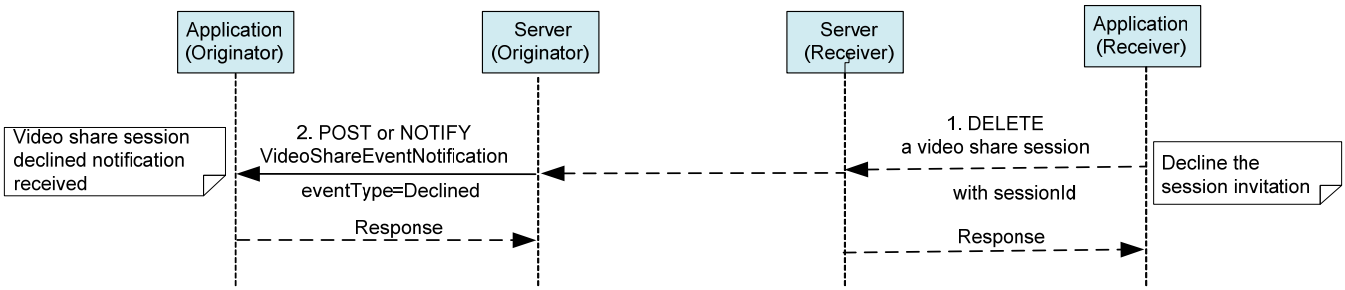

Figure 6 Declining a 1-1 video share session invitation

Outline of the flows:

An application of the Originator has created a video share session resource triggering a video share invitation sent to the Receiver (Refer to step 1 and step 2 in 5.3.2). Subsequently:

1. The application of the Receiver declines the video share session invitation using the DELETE method on the session resource including the sessionId.
2. The application of the Originator receives a notification containing the VideoShareEventNotification data structure indicating that the Receiver has declined the invitation.

Note that declining a session only works before the Receiver has accepted the video share invitation. After that, the DELETE method leads to an existing session to be terminated.

5.3.3.3 1-1 video share failed

The figure below shows a scenario for video share failed.

The resources:

- Application provided

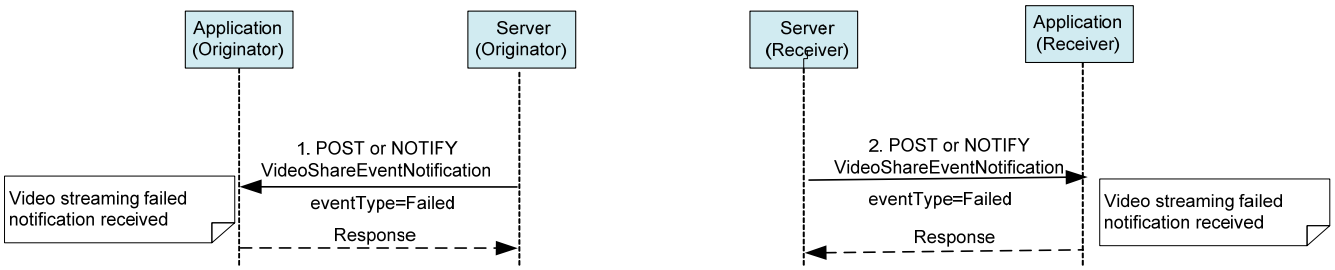

Figure 7 Video share failed

Outline of the flows:

1. When error occurs during the video share session (e.g. after step 4 and before step 5 in 5.3.2), the application of the Originator receives a notification containing the VideoShareEventNotification data structure indicating that the video share has failed.
2. The application of the Receiver also receives a notification containing the VideoShareEventNotification data structure indicating that the video share has failed.

6. Detailed specification of the resources

The following applies to all resources defined in this specification regardless of the representation format (i.e. XML, JSON, application/x-www-form-urlencoded):

- Reserved characters in URL variables (parts of a URL denoted below by a name in curly brackets) MUST be percent-encoded according to [RFC3986]. Note that this always applies, no matter whether the URL is used as a Request URL or inside the representation of a resource (such as in "resourceURL" and "link" elements).

- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an MSISDN, it MUST be defined as a global number according to [RFC3966] (e.g. tel:+19585550100).The use of characters other than digits and the leading "+" sign SHOULD be avoided in order to ensure uniqueness of the resource URL. This applies regardless of whether the user identifier appears in a URL variable or in a parameter in the body of an HTTP message.

- If a user identifier (e.g. address, userId, etc.) of type anyURI is in the form of a SIP URI, it MUST be defined according to [RFC3261].

- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an Anonymous Customer Reference (ACR), it MUST be defined according to [IETF_ACR_draft], i.e. it MUST include the protocol prefix 'acr:' followed by the ACR.

    - The ACR 'Authorization' is a supported reserved keyword, and MUST NOT be assigned as an ACR to any particular end user. See G.1.2 for details regarding the use of this reserved keyword.

- For requests and responses that have a body, the following applies: in the requests received, the server SHALL support JSON and XML encoding of the parameters in the body, and MAY support application/x-www-form-urlencoded parameters in the body. The Server SHALL return either JSON or XML encoded parameters in the response body, according to the result of the content type negotiation as specified in [REST_NetAPI_Common]. In notifications to the Client, the server SHALL use either XML or JSON encoding, depending on which format the client has specified in the related subscription. The generation and handling of the JSON representations SHALL follow the rules for JSON encoding in HTTP Requests/Responses as specified in [REST_ NetAPI_Common].

6.1 Resource: All subscriptions to video share notifications

The resource used is:

http://{serverRoot}/videoshare/{apiVersion}/{userId}/subscriptions

This resource is used to manage subscriptions to video share notifications. Note that there is one subscription per client instance.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.1.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI

apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.1.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Video Share API, see section 7.

6.1.3 GET

This operation is used for reading the list of active video share notification subscriptions.

6.1.3.1 Example 1: Reading all active video share notification subscriptions (Informative)

6.1.3.1.1 Request

```
GET /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
```

6.1.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSubscriptionList xmlns:vs="urn:oma:xml:rest:netapi:videoshare:1">
    <videoShareNotificationSubscription>
        <callbackReference>
            <notifyURL>http://application.example.com/videoshare/notifications/77777</notifyURL>
            <callbackData>abcd</callbackData>
        </callbackReference>
        <duration>7200</duration>
        <clientCorrelator>12345</clientCorrelator>
        <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
    </videoShareNotificationSubscription>
    <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions</resourceURL>
</vs:videoShareSubscriptionList>
```

6.1.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.1.5 POST

This operation is used to create a new subscription for video share notifications.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.1.5.1 Example 1: Creating a new subscription to video share notifications using tel URI (Informative)

6.1.5.1.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/ HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareNotificationSubscription xmlns:vs="urn:oma:xml:rest:netapi:videoshare:1">
    <callbackReference>
        <notifyURL>http://application.example.com/videoshare/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
</vs:videoShareNotificationSubscription>
```

6.1.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareNotificationSubscription xmlns:vs="urn:oma:xml:rest:netapi:videoshare:1">
    <callbackReference>
        <notifyURL>http://application.example.com/videoshare/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
   <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</vs:videoShareNotificationSubscription>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.1.5.2.2).

6.1.5.2 Example 2: Creating a new subscription to video share notifications using ACR (Informative)

6.1.5.2.1 Request

```
POST /exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/ HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareNotificationSubscription xmlns:vs="urn:oma:xml:rest:netapi:videoshare:1">
    <callbackReference>
        <notifyURL>http://application.example.com/videoshare/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
</vs:videoShareNotificationSubscription>
```

6.1.5.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/sub001
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/sub001</resourceURL>
</common:resourceReference>
```

6.1.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.2 Resource: Individual subscription to video share event notifications

The resource used is:

http://{serverRoot}/videoshare/{apiVersion}/{userId}/subscriptions/{subscriptionId}

This resource represents an individual subscription to video share notifications.

6.2.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use. The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts. Examples: tel:+19585550100, acr:pseudonym123
subscriptionId	Identifier of the subscription

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.2.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Video Share API, see section 7.

6.2.3 GET

This operation is used for reading an individual subscription.

6.2.3.1 Example: Reading an individual subscription (Informative)

This example shows also an alternative way to indicate desired content type in response from the server, by using URL query parameter "?resFormat" which is described in [REST_NetAPI_Common].

6.2.3.1.1 Request

```
GET /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001?resFormat=XML HTTP/1.1
Host: example.com
```

6.2.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareNotificationSubscription xmlns:vs="urn:oma:xml:rest:netapi:videoshare:1">
    <callbackReference>
        <notifyURL>http://application.example.com/videoshare/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
```

```
    <clientCorrelator>12345</clientCorrelator>
    <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</vs:videoShareNotificationSubscription>
```

6.2.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC2616].

6.2.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC2616].

6.2.6 DELETE

This operation is used to cancel a subscription and to stop corresponding notifications.

6.2.6.1 Example: Cancelling a subscription (Informative)

6.2.6.1.1 Request

```
DELETE /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.2.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jun 2010 17:51:59 GMT
```

6.3 Resource: All 1-1 video share sessions

The resource used is:

http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions

This resource represents the active 1-1 video share sessions for a particular user.

6.3.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path.

	Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use. The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.3.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Video Share API, see section 7.

6.3.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.3.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.3.5 POST

This operation is used to create a new 1-1 video share session.

6.3.5.1 Example 1: Creating a new 1-1 video share session with mediaURL for recorded video (no CS call related) (Informative)

6.3.5.1.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName >
  <receiverAddress>tel:+19585550101</receiverAddress>
  <receiverName>Bob</receiverName>
```

```
  <mediaURL>http://myvideos.com/recorded/holidays/20110501/file1/</mediaURL>
  <clientCorrelator>12345</clientCorrelator>
</vs:videoShareSessionInformation>
```

6.3.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName>
  <receiverAddress>tel:+19585550101</receiverAddress>
  <receiverName>Bob</receiverName>
  <mediaURL>http://myvideos.com/recorded/holidays/20110501/file1/</mediaURL>
  <status>Initial</status>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</vs:videoShareSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.1.5.2.2).

6.3.5.2 Example 2: Creating a new 1-1 video share session with recorded video file content (CS call related) (Informative)

6.3.5.2.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
Accept: application/xml
Host: example.com
MIME-Version: 1.0

--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName >
  <callObjectRef>http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001</callObjectRef>
  <clientCorrelator>12345</clientCorrelator>
</vs:videoShareSessionInformation>

--===============123456==
```

```
Content-Disposition: form-data; name=" attachments ";filename="file1"
Content-Type: video/H263-2000
Content-Transfer-Encoding: binary
Content-Length: [length of video file]


 ...binary video file...

--===============123456==--
```

6.3.5.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName>
  <receiverAddress>tel:+19585550101</receiverAddress>
  <callObjectRef>http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001</callObjectRef>
  <receiverName>Bob</receiverName>
  <status>Initial</status>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</vs:videoShareSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.1.5.2.2).

6.3.5.3 Example 3: Creating a new 1-1 video share session with live video (no CS call related) (Informative)

6.3.5.3.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
<originatorAddress>tel:+19585550100</originatorAddress>
 <originatorName>Alice</originatorName >
 <receiverAddress>tel:+19585550101</receiverAddress>
 <receiverName>Bob</receiverName>
 <mediaInformation>
   <mediaType>Video</mediaType >
   <transportProtocol>RTP/AVP</transportProtocol>
```

```
   <mediaFormats>
     <encodingName>H263-2000</encodingName>
     <clockRate>90000</clockRate>
     <fmtp>profile=0; level=45</fmtp>
   </mediaFormats>
   <mediaFormats>
     <encodingName>MP4V-ES</encodingName>
   </mediaFormats>
   <bandwidth>54</bandwidth>
   <bwType>AS</bwType>
   <frameRate>8</frameRate>
 </mediaInformation>
 <clientCorrelator>12345</clientCorrelator>
</vs:videoShareSessionInformation>
```

6.3.5.3.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
<originatorAddress>tel:+19585550100</originatorAddress>
 <originatorName>Alice</originatorName >
 <receiverAddress>tel:+19585550101</receiverAddress>
 <receiverName>Bob</receiverName>
<mediaInformation>
   <mediaType>Video</mediaType >
   <transportProtocol>RTP/AVP</transportProtocol>
   <mediaFormats>
     <fmt>96</fmt>
     <encodingName>H263-2000</encodingName>
     <clockRate>90000</clockRate>
     <fmtp>profile=0; level=45</fmtp>
   </mediaFormats>
   <mediaFormats>
      <fmt>97</fmt>
     <encodingName>MP4V-ES</encodingName>
   </mediaFormats>
   <bandwidth>54</bandwidth>
   <bwType>AS</bwType>
   <frameRate>8</frameRate>
</mediaInformation>
<status>Initial</status>
 <clientCorrelator>12345</clientCorrelator>
<resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</vs:videoShareSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.1.5.2.2).

6.3.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.4 Resource: Individual 1-1 video share session

The resource used is:

http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}

This resource represents a 1-1 video share session.

6.4.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use. The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts. Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the session

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.4.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Video Share API, see section 7.

6.4.3 GET

This operation is used to retrieve video share session information.

6.4.3.1 Example 1: Retrieving 1-1 video share session information (Informative)

6.4.3.1.1 Request

```
GET /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName>
  <receiverAddress>tel:+19585550101</receiverAddress>
  <callObjectRef>http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001</callObjectRef>
  <receiverName>Bob</receiverName>
  <mediaURL>http://myvideos.com/recorded/holidays/20110501/file1/</mediaURL>
  <status>Connected</status>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</vs:videoShareSessionInformation>
```

6.4.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC2616].

6.4.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC2616].

6.4.6 DELETE

This operation ends the 1-1 video share session.

It is used by an application of the Originator to cancel a 1-1 video share session before the Receiver has accepted the session invitation.

It is used by an application of the Receiver to decline a 1-1 video share session when the session invitation has been received.

It is used by an application of the Originator or an application of the Receiver to terminate a 1-1 video share session after the session has been accepted.

6.4.6.1 Example: Terminating a 1-1 video share session (Informative)

6.4.6.1.1 Request

```
DELETE /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.6.1.2 Response

HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT

6.5 Resource: Individual 1-1 video share session status

The resource used is:

http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}/status

This resource represents the status of the 1-1 video session and is used for accepting a 1-1 video share invitation, by means of updating the status.

6.5.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use. The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts. Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the session

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.5.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Video Share API, see section 7.

6.5.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.5.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.5.5 POST

This operation is used for accepting a 1-1 video share invitation, by means of updating the status.

6.5.5.1 Example1: Accepting a 1-1 video share invitation with accepted media information (Informative)

6.5.5.1.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xmlHost: example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:receiverSessionStatus xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
   <status>Connected</status>
   <mediaInformation>
      <mediaType>Video</mediaType >
      <transportProtocol>RTP/AVP</transportProtocol>
      <mediaFormats>
        <fmt>96</fmt>
        <encodingName>H263-2000</encodingName>
        <clockRate>90000</clockRate>
        <fmtp>profile=0; level=45</fmtp>
      </mediaFormats>
     <bandwidth>54</bandwidth>
     <bwType>AS</bwType>
     <frameRate>8</frameRate>
  </mediaInformation>
</vs:receiverSessionStatus>
```

6.5.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

6.5.5.2 Example2: Accepting a 1-1 video share invitation without accepted media information (Informative)

6.5.5.2.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:receiverSessionStatus xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <status>Connected</status>
```

```
</vs:receiverSessionStatus>
```

6.5.5.2.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

6.5.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.6 Resource: Client notification about 1-1 video share session invitations

This resource is a callback URL provided by the client for notification about 1-1 video share session invitations. The RESTful Video Share API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.6.5.

The following table gives detailed information about video share session invitation notification. It is also outlined which Participant receives notifications of a particular type, whether a response is needed, and which links to other resources are contained in that notification.

In the "Notification sent to" column, the following values can occur:

- Originator: the Originator of the video share session
- Receiver: one individual Receiver in the video share session at a time
- All: Receiver and Originator of the video share session at once

Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: http://{serverRoot}/video share/{apiVersion}/{userId}/sessions
SessionInvitationNotification	Receiver	decline accept	VideoShareSessionInformation ReceiverSessionStatus	/{sessionId} /{sessionId}/status

Table 2: 1-1 video share session invitation notification

The application of Receiver can accept the request by updating the status, which is addressed by the URL passed in the “href” attribute of the “link” element with rel=”ReceiverSessionStatus”.

Typically, this is http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}/status.

The application of Receiver can decline the request by sending a DELETE request to one the URL passed in the “href” attribute of the “link” element with rel=”VideoShareSessionInformation”.

Typically, this is http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions/{sessionId}

If the application of Receiver fails to react within a time interval defined by service policies, the session invitation will time out. In case of a 1-1 session, this means that the session will terminate.

6.6.1 Request URL variables

Client provided if any.

6.6.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

6.6.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC2616].

6.6.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC2616].

6.6.5 POST

This operation is used to notify the client about video share session invitations.

6.6.5.1 Example 1: Notify a client about 1-1 video share session invitations (no CS call related) (Informative)

6.6.5.1.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1

Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com
```

```
<?xml version="1.0" encoding="UTF-8"?>
<vs:sessionInvitationNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
  <link rel="ReceiverSessionstatus"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status"/>
  <link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName>
  <receiverAddress>tel:+19585550101</receiverAddress>
  <receiverName>Bob</receiverName>
   <mediaInformation>
    <mediaType>Video</mediaType>
    <transportProtocol>RTP/AVP</transportProtocol>
    <mediaFormats>
        <fmt>96</fmt>
        <encodingName>H263-2000</encodingName>
        <clockRate>90000</clockRate>
     <fmtp>profile=0; level=45</fmtp>
    </mediaFormats>
   <mediaFormats>
     <fmt>97</fmt>
     <encodingName>MP4V-ES</encodingName>
   </mediaFormats>
    <bandwidth>54</bandwidth>
    <bwType>AS</bwType>
    <frameRate>8</frameRate>
  </mediaInformation>
</vs:sessionInvitationNotification>
```

6.6.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.6.5.2 Example 2: Notify a client about 1-1 video share session invitations (CS call related) (Informative)

6.6.5.2.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1

Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:sessionInvitationNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
<link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
<link rel="ReceiverSessionstatus"
```

```
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status"/>
<link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
<originatorAddress>tel:+19585550100</originatorAddress>
<originatorName>Alice</originatorName>
<receiverAddress>tel:+19585550101</receiverAddress>
<callObjectRef>http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001</callObjectRef>
<receiverName>Bob</receiverName>
 <mediaInformation>
   <mediaType>Video</mediaType>
   <transportProtocol>RTP/AVP</transportProtocol>
  <mediaFormats>
        <fmt>96</fmt>
        <encodingName>H263-2000</encodingName>
        <clockRate>90000</clockRate>
     <fmtp>profile=0; level=45</fmtp>
    </mediaFormats>
   <mediaFormats>
      <fmt>97</fmt>
      <encodingName>MP4V-ES</encodingName>
   </mediaFormats>
   <bandwidth>54</bandwidth>
   <bwType>AS</bwType>
  <frameRate>8</frameRate>
 </mediaInformation>
</vs:sessionInvitationNotification>
```

6.6.5.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.6.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.7 Resource: Client notification about 1-1 video share session acceptance

This resource is a callback URL provided by the client for notification about 1-1 video share session acceptance. The RESTful Video Share API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.7.5.

The following table gives detailed information about 1-1 video share session acceptance notification. It is also outlined which PParticipant receives notifications of a particular type, whether a response is needed, and which links to other resources are contained in that notification.

In the "Notification sent to" column, the following values can occur:

- Originator: the Originator of the video share session
- Receiver: one individual Receiver in the video share session at a time
- All: Receiver and Originator of the video share session at once

Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions
SessionAcceptanceNotification	Originator	n/a	VideoShareSessionInformation	/{sessionId}

Table 3: 1-1 video share session acceptance notification

6.7.1 Request URL variables

Client provided if any.

6.7.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

6.7.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.7.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.7.5 POST

This operation is used to notify the client about video share session acceptance.

6.7.5.1 Example 1: Notify a client about the acceptance of 1-1 video share session with recorded video (no CS call related) (Informative)

6.7.5.1.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1

Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:sessionAcceptanceNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <receiverAddress>tel:+19585550101</receiverAddress>
  <receiverName>Bob</receiverName>
  <status>Connected</status>

  <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
  <link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
</vs:sessionAcceptanceNotification>
```

6.7.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.7.5.2 Example 2: Notify a client about the acceptance of 1-1 video share session with recorded video (CS call related) (Informative)

6.7.5.2.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1

Content-Type: application/xml
Content-Length: nnnn
Accept: application/xmlHost: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:sessionAcceptanceNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <receiverAddress>tel:+19585550101</receiverAddress>
  <receiverName>Bob</receiverName>
  <callObjectRef>http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001</callObjectRef>
  <status>Connected</status>
  <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
<link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
</vs:sessionAcceptanceNotification>
```

6.7.5.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.7.5.3 Example 3: Notify a client about the acceptance of 1-1 video share session with live video (no CS call related) (Informative)

6.7.5.3.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1

Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:sessionAcceptanceNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
 <receiverAddress>tel:+19585550101</receiverAddress>
 <receiverName>Bob</receiverName>
 <status>Connected</status>
 <mediaURL>http://application.example.com/userId/tel%3A%2B19585550101/20110728175159/file1</mediaURL>
 <mediaInformation>
   <mediaType>Video</mediaType >
   <transportProtocol>RTP/AVP</transportProtocol>
   <mediaFormats>
    <fmt>96</fmt>
    <encodingName>H263-2000</encodingName>
    <clockRate>90000</clockRate>
    <fmtp>profile=0; level=45</fmtp>
   </mediaFormats>
   <bandwidth>54</bandwidth>
   <bwType>AS</bwType>
   <frameRate>8</frameRate>
 </mediaInformation>
 <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
<link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
</vs:sessionAcceptanceNotification>
```

6.7.5.3.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.7.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC2616].

6.8 Resource: Client notification about 1-1 video share events

This resource is a callback URL provided by the client for notification about 1-1 video share events. The RESTful Video Share API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.8.5.

The following table gives an overview of video share event notification. It is also outlined which Participants receive notifications of a particular type, whether a response is needed, and which links to other resources are contained in that notification.

In the "Notification sent to" column, the following values can occur:

- Originator: the Originator of the video share session
- Receiver: one individual Receiver in the video share session at a time
- All: Receiver and Originator of the video share session at once

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions
Declined	VideoShareEventNotification	Originator	n/a	VideoShareSessionInformation	/{sessionId}
SessionCancelled	VideoShareEventNotification	Receiver	n/a	VideoShareSessionInformation	/{sessionId}
SessionEnded	VideoShareEventNotification	All	n/a	VideoShareSessionInformation	/{sessionId}
Failed	VideoShareEventNotification	All	n/a	VideoShareSessionInformation	/{sessionId}

Table 4: 1-1 video share event notification

6.8.1 Request URL variables

Client provided if any.

6.8.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

6.8.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.8.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.8.5 POST

This operation is used to notify the client about video share events.

6.8.5.1 Example 1: Notify a client about 1-1 video share event (ended) (Informative)

6.8.5.1.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareEventNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <eventType>SessionEnded</eventType>
  <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
<link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
</vs:videoShareEventNotification>
```

6.8.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.8.5.2 Example 2: Notify a client about 1-1 video share event (declined) (Informative)

6.8.5.2.1 Request

```
POST /notifications/77777 HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareEventNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <eventType>Declined</eventType>
  <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
<link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
</vs:videoShareEventNotification>
```

6.8.5.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.8.5.3 Example 3: Notify a client about 1-1 video share event (cancelled) (Informative)

6.8.5.3.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareEventNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <eventType>SessionCancelled</eventType>
  <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
<link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
</vs:videoShareEventNotification>
```

6.8.5.3.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.8.5.4 Example 4: Notify a client about 1-1 video share event (failed)(Informative)

6.8.5.4.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareEventNotification xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <eventType>Failed</eventType>
  <link rel="VideoShareSessionInformation"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001"/>
<link rel="VideoShareNotificationSubscription"
href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001"/>
</vs:videoShareEventNotification>
```

6.8.5.4.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.8.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.9 Resource: Client notification about subscription cancellations

This resource is a callback URL provided by the client for notification about subscription cancellations, which are usually due to the subscription expiring. The RESTful Video Share API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.9.5 .

The notification is sent by the server to the Participant to whom the cancelled subscription belongs.

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}/sessions

n/a	SubscriptionCancellationNotification	Participant	n/a	VideoShareNotificationSubscription	/subscriptions/{subscriptionId}

6.9.1 Request URL variables

Client provided if any.

6.9.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Video Share API, see section 7.

6.9.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.9.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.9.5 POST

This operation is used to notify the client about subscription cancellation.

6.9.5.1 Example: Notify a client about subscription cancellation (Informative)

6.9.5.1.1 Request

```
POST /videoshare/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<vs:subscriptionCancellationNotification xmlns:vs="urn:oma:xml:rest:netapi:videoshare:1">
  <callbackData>abcd</callbackData>
  <link rel="VideoShareNotificationSubscription"
    href="http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001"/>
</vs:subscriptionCancellationNotification >
```

6.9.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.9.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

7. Fault definitions

7.1 Service Exceptions

For common Service Exceptions refer to [REST_NetAPI_Common]. The following additional Service Exception codes are defined for the RESTful Video Share API.

7.1.1 SVC1002: CS call not existing for Video Share

Name	Description
MessageID	SVC1002
Text	CS call object reference not existing for Video Share
Variables	None
HTTP status code(s)	400 Bad request

7.2 Policy Exceptions

For common Policy Exceptions refer to [REST_NetAPI_Common].

No policy exceptions are defined for the RESTful Video Share API in this release.

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS-REST_NetAPI_VideoShare-V1_0	10 May 2011	All	Baseline
	17 May 2011	5.1	CRs implemented - OMA-ARC-REST-NetAPI-2011-0014-CR_VideoShare_ResourceSummary_originating_sid - OMA-ARC-REST-NetAPI-2011-0015-CR_CR_VideoShare_ResourceSummary_originating_side (Note : CR0014 and CR 0015 are identical)
	15 Jun 2011	5, 5.1, Appendix E	CRs implemented - OMA-ARC-REST-NetAPI-2011-0025R03-CR_VideoShare_Resources
	02 Aug 2011	5.1	CRs implemented - OMA-ARC-REST-NetAPI-2011-0130R02-CR_VideoShare_Resources_alignment_with_new_resource_model
	08 Oct 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0183R03-CR_VideoShare_DataTypes_with_tel_URI_and_Notif_channel_changes
	31 Oct 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0286-CR_VideoShare_ACR - OMA-ARC-REST-NetAPI-2011-0290R01-CR_VideoShare_DataTypes_update_SequenceDiagrams
	10 Nov 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0347R01-CR_VideoShare_Sequece_diagrams_update - OMA-ARC-REST-NetAPI-2011-0348R01-CR_VideoShare_DateType_improvements
	27 Nov 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0388R02-CR_videoshare_detailed_specification_of_resources - OMA-ARC-REST-NetAPI-2011-0392R02-CR_VideoShare_media_datatTypes
	01 Dec 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0417-CR_VideoShare_Appendix_G - OMA-ARC-REST-NetAPI-2011-0420R01-CR_VideoShare_SCR - OMA-ARC-REST-NetAPI-2011-0428R03-CR_VideoShare_Appendix_C - Change "acr:authorization" to "acr:Authorization"
	06 Feb 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0042-CR_VideoShare_CONR_editorial_changes
	20 Feb 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0071-CR_VideoShare_CONR_link_examples
	23 Feb 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0072-CR_VideoShare_CONR_appendixC - OMA-ARC-REST-NetAPI-2012-0081R02-CR_VideoShare_CONR_resolution_more

Document Identifier	Date	Sections	Description
	06 Mar 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0091-CR_VideoShare_CONR_TS_resolutions
	19 Mar 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0097-CR_VideoShare_TS_more_fix
	20 Mar 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0111-CR_VideoShare_DataType_Improvements
	27 Mar 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0115R01-CR_VideoShare_TS_Exceptions_Notifications_with_TTL
	28 Mar 2012	P2	Corrected copyright year to 2012 Editorial changes
Candidate Version OMA-TS-REST_NetAPI_VideoShare-V1_0	10 Apr 2012	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2012-0163-INP_REST_NetAPI_VideoShare_1_0_ERP_and_ETR_for_Candidate_Approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

B.1 SCR for REST.VideoShare Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-SUPPORT-S-001-M	Support for the RESTful Video Share API	5, 6	
REST-VIDEOSHARE-SUPPORT-S-002-M	Support for the XML request & response format	6	
REST-VIDEOSHARE-SUPPORT-S-003-M	Support for the JSON request & response format	6	
REST-VIDEOSHARE-SUPPORT-S-004-O	Support for the application/x-www-form-urlencoded format	Appendix C	

B.1.1 SCR for REST.VideoShare.Subscriptions Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-SUBSCR-S-001-M	Support for subscriptions to video share notifications	6.1	
REST-VIDEOSHARE-SUBSCR-S-002-O	Read the list of active video share notification subscriptions – GET	6.1.3	
REST-VIDEOSHARE-SUBSCR-S-003-M	Create new subscription to video share notifications – POST (XML or JSON)	6.1.5	
REST-VIDEOSHARE-SUBSCR-S-004-O	Create new subscription to video share notifications – POST (application/x-www-form-urlencoded)	C.1	

B.1.2 SCR for REST.VideoShare.Individual.Subscription Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-IND-SUBSCR-S-001-M	Support for access to an individual subscription to video share notifications	6.2	
REST-VIDEOSHARE-IND-SUBSCR-S-002-O	Read an individual video share notification subscription – GET	6.2.3	

Item	Function	Reference	Requirement
REST-VIDEOSHARE-IND-SUBSCR-S-003-M	Cancel subscription and stop corresponding notifications – DELETE	6.2.6	

B.1.3 SCR for REST.VideoShare.Sessions Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-SESS-S-001-M	Support for 1-1 video share sessions	6.3	
REST-VIDEOSHARE-SESS-S-002-M	Create a new 1-1 video share session – POST(XML or JSON)	6.3.5	
REST-VIDEOSHARE-SESS-S-003-O	Create a new 1-1 video share session – POST(application/x-www-form-urlencoded)	C.2	

B.1.4 SCR for REST.VideoShare.Individual.Session Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-IND-SESS-S-001-M	Support for individual 1-1 video share sessions	6.4	
REST-VIDEOSHARE-IND-SESS-S-002-O	Retrieve a 1-1 video share session information – GET	6.4.3	
REST-VIDEOSHARE-IND-SESS-S-003-M	Cancel invitation/ Decline Invitation/ Terminate a 1-1 video share session – DELETE	6.4.6	

B.1.5 SCR for REST.VideoShare.Individual.Session.Status Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-IND-SESS-STAT-S-001-M	Support for acceptance of the 1-1 video share session	6.5	
REST-VIDEOSHARE-IND-SESS-STAT-S-002-M	Accept a 1-1 video share session invitation – POST (XML or JSON)	6.5.5	
REST-VIDEOSHARE-IND-SESS-STAT-S-003-O	Accept a 1-1 video share session invitation – POST(application/x-www-form-urlencoded)	C.3	

B.1.6 SCR for REST.VideoShare.Session.Invitation.Notifications Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-INVITE-NOTIF-S-001-M	Support for notifications about 1-1 video share session invitations	6.6	
REST-VIDEOSHARE-INVITE-NOTIF-S-002-M	1-1 Video share session invitation notifications – POST (XML or JSON)	6.6.5	

B.1.7 SCR for REST.VideoShare.Session.Acceptance.Notifications Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-ACCEPT-NOTIF-S-001-M	Support for notifications about 1-1 video share session acceptance	6.7	
REST-VIDEOSHARE-ACCEPT-NOTIF-S-002-M	1-1 Video share session acceptance notifications – POST (XML or JSON)	6.7.5	

B.1.8 SCR for REST.VideoShare.Events.Notifications Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-EVENT-NOTIF-S-001-M	Support for notifications about 1-1 video share events	6.8	
REST-VIDEOSHARE-EVENT-NOTIF-S-002-M	1-1 Video share event notifications – POST (XML or JSON)	6.8.5	

B.1.9 SCR for REST.VideoShare.SubscriptionCancellation Server

Item	Function	Reference	Requirement
REST-VIDEOSHARE-NOTIF-SUBCXL-S-001-M	Support for notifications about subscription cancellation	6.9	
REST-VIDEOSHARE-NOTIF-SUBCXL-S-002-M	Subscription cancellation notifications – POST (XML or JSON)	6.9.5	

Appendix C. Application/x-www-form-urlencoded Request Format for POST Operations (Normative)

This section defines a format for the RESTful Video Share API requests where the body of the request is encoded using the application/x-www-form-urlencoded MIME type.

Note: only the request body is encoded as application/x-www-form-urlencoded, the response is still encoded as XML or JSON depending on the preference of the client and the capabilities of the server. Names and values MUST follow the application/x-www-form-urlencoded character escaping rules from [W3C_URLENC].

The encoding is defined below for the following Video Share REST operations which are based on POST requests:

- Creating a new subscription to video share notifications
- Creating a new 1-1 video share session
- Accepting a 1-1 video share session invitation

C.1 Creating a new subscription to video share notifications

This operation is used to create a new subscription to video share notifications. See section 6.1.5.

The notifyURL either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

The request parameters are as follows:

Name	Type/Values	Optional	Description
notifyURL	xsd:anyURI	No	Notification endpoint definition. For the use of Client-side Notification URLs and Server-side Notification URLs in this parameter, see sections 6.1 and 6.1.5.
callbackData	xsd:string	Yes	Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications.
notificationFormat	common:NotificationFormat	Yes	Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}. Default: XML
duration	xsd:int	Yes	Period of time (in seconds) notifications are provided for. If set to “0” (zero), a default duration time, which is specified by the service policy, will be used. If the parameter is omitted, the notifications will continue until the maximum duration time, which is specified by the service policy, unless the notifications are stopped by deletion of subscription for notifications. This element MAY be given by the client during resource creation in order to signal the desired lifetime of the subscription. The server SHOULD return in this element the period of time for which the subscription will still be valid.

clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of "201 Created".

C.1.1 Example: Creating a new subscription to video share notifications using tel URI (Informative)

C.1.1.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

notifyURL=http%3A%2F%2Fapplication.example.com%2Fvideoshare%2Fnotifications%2F77777&
callbackData=abcd&
duration=7200&
clientCorrelator=12345
```

C.1.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareNotificationSubscription xmlns:vs="urn:oma:xml:rest:netapi:videoshare:1">
    <callbackReference>
        <notifyURL>http://application.example.com/videoshare/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
  <duration>7200</duration>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</vs:videoShareNotificationSubscription>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section C.1.2.2).

C.1.2 Example: Creating a new subscription to video share notifications using ACR (Informative)

C.1.2.1 Request

```
POST /exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/ HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

notifyURL=http%3A%2F%2Fapplication.example.com%2Fvideoshare%2Fnotifications%2F77777&
callbackData=abcd&
duration=7200&
clientCorrelator=12345
```

C.1.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/sub001
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/sub001</resourceURL>
</common:resourceReference>
```

C.2 Creating a new 1-1 video share session

This operation is used to create a new 1-1 video share session. See section 6.3.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
originatorAddress	xsd:anyURI	No	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator
originatorName	xsd:string	Yes	Name of the Originator
receiverAddress	xsd:anyURI	Yes	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Receiver. It SHALL be present in request bodies during resource creation in case of video sharing without CS voice call. It SHALL not be present in request bodies during resource creation in case of video sharing with CS voice call. The server can get the receiverAddress

			using the callObjectRef received in the request bodies during resource creation in case of video sharing with CS voice call.
callObjectRef	xsd:anyURI	Yes	Reference to the CS voice object (to which the Receiver is linked). It SHALL be present in request bodies during resource creation in case of video sharing with CS voice call. It SHALL not be present in request bodies during resource creation in case of video sharing without CS voice call.
receiverName	xsd:string	Yes	Name of the Receiver. MAY be present when receiverAddress is present.
mediaURL	xsd:anyURI	Yes	The video media URL from where the video content can be retrieved. It SHALL not be present in the POST request during resource creation if liveVideo is set to true. If it is present in the POST operation during resource creation, the server could fetch the video content using this URL. If it is not present in the POST request during resource creation and the liveVideo is not set to true (not live video), the video content is included in the HTTP body. The HTTP body can be represented as multipart/form-data entity bodies, where the first entry of the form is the root element and the second entry of the form is the video content.
size	xsd: unsignedLong	Yes	The size of the video file in octets
mediaType	MediaType [0…unbounded]	Yes	Type of the media a (e.g. “Audio” or “Video”). See description of “m=” field in [RFC4566].
transportProtocol	xsd:string [0…unbounded]	Yes	The transport protocol of the media (e.g. “RTP/AVP”, “RTP/SAVP”, ”udp”). See description of “m=” field in [RFC4566].
fmt	xsd:string [0…unbounded]	Yes	When “transportProtocol” is "RTP/AVP" or "RTP/SAVP”, it contains static or dynamic RTP payload type number. When static RTP payload type number is used, it SHALL be present in the request bodies during the session resource creation and the rest of the elements in this structure are not needed. The dynamic RTP payload type number is generated either by the server or by the client. If generated by the client, it SHALL be present in the request bodies during the session resource creation; if generated by

			the server, it SHALL not be present in the request bodies during the session resource creation. When dynamic RTP payload type number is used, at least “encodingName” SHALL be present. See description of “m=” field in [RFC4566].
encodingName	xsd:string [0…unbounded]	Yes	The encoding name of the RTP payload type. See description of “a=rtpmap” field in [RFC4566]. It SHALL use media subtype (e.g., “AMR” for audio, “H263-2000” for video.) when “RTP/AVP" or "RTP/SAVP" transport protocol is used, see IANA for registered media subtypes for audio and video.
clockRate	xsd:unsignedLong [0…unbounded]	Yes	Number of samples per second, see description of “a=rtpmap” field in [RFC4566].
encodingParameter	xsd:string [0…unbounded]	Yes	List of encoding parameters for the RTP payload type, see description of “a=rtpmap” field in [RFC4566].
fmtp	xsd:string [0…unbounded]	Yes	Format specific parameters of “a=fmtp” as defined in [RFC4566].
bandwidth	xsd:unsignedLong [0…unbounded]	Yes	Band width in kilobits per second. See description of “b=” field in [RFC4566].
bwType	xsd:string [0…unbounded]	Yes	Currently only CT" (Conference Total) and "AS" (Application Specific) are supported. See description of “b=” field in [RFC4566].
frameRate	xsd:decimal [0…unbounded]	Yes	The maximum video frame rate in frames/sec, defined only for video media. See description of “a=framerate” field in [RFC4566].
pTime	xsd:unsignedInt [0…unbounded]	Yes	The length of time in milliseconds represented by the media in a packet. It is probably only meaningful for audio data, but may be used with other media types if it makes sense. See description of “a=ptime” field in [RFC4566].
maxPTime	xsd:unsignedInt [0…unbounded]	Yes	This gives the maximum amount of media that can be encapsulated in each packet, expressed as time in milliseconds. For frame-based codecs, the time SHOULD be an integer multiple of the frame size. It is probably only meaningful for audio data, but may be used with other media types if it makes sense. See description of “a=maxptime” field in [RFC4566].
attributeName	xsd:string	Yes	Name of the attribute

	[[0…unbounded]		
attributeValue	xsd:string [0…unbounded]	Choice	The value of the attribute
<any element>	< type is defined in a schema implementing the element> [0…unbounded]	Choice	The value of the attribute. Note that element 'any element' can be any element from any other namespace (schema) than the target namespace. Type of such element is defined by the schema implementing the element. In XML implementations, element "any" must be qualified with the namespace prefix.
liveVideo	xsd:boolean	Yes	Indicates whether the video is live video (true) or video clip (false). This element MUST be present and set to "true" if the video is live. Default value is 'false.'
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of "201 Created".

C.2.1 Example 1: Creating a new 1-1 video share session with mediaURL for recorded video (no CS call related) (Informative)

C.2.1.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

 originatorAddress=tel%3A%2B19585550100&
 originatorName=Alice&
receiverAddress=tel%3A%2B19585550101&
receiverName=Bob&
mediaURL=http%3A%2F%2Fmyvideos.com%2Frecorded%2Fholidays%2F20110501%2Ffile1/&
clientCorrelator=12345
```

C.2.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName>
  <receiverAddress>tel:+19585550101</receiverAddress>
  <receiverName>Bob</receiverName>
  <mediaURL>http://myvideos.com/recorded/holidays/20110501/file1/</mediaURL>
  <status>Initial</status>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</vs:videoShareSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section C.1.2.2).

C.2.2 Example 2: Creating a new 1-1 video share session with recorded video file content (CS call related) (Informative)

C.2.2.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
Accept: application/xml
Host: example.com
MIME-Version: 1.0

--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

originatorAddress=tel%3A%2B19585550100&
originatorName=Alice&
callObjectRef=http%3A%2F%2Fexample.com%2FexampleAPI%2Fcall%2Ftel%3A%2B19585550101%2Fsessions%2FcallSess001&
clientCorrelator=12345

--===============123456==
Content-Disposition: form-data; name=” attachments ”;filename=”file1”
Content-Type: video/H263-2000
Content-Transfer-Encoding: binary
Content-Length: [length of video file]


 ...binary video file...
```

```
--===============123456==--
```

C.2.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Alice</originatorName>
  <receiverAddress>tel:+19585550101</receiverAddress>
  <callObjectRef>http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001</callObjectRef>
  <receiverName>Bob</receiverName>
  <status>Initial</status>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</vs:videoShareSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section C.1.2.2).

.

C.2.3 Example 3: Creating a new 1-1 video share session with live video (no CS call related) (Informative)

C.2.3.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

originatorAddress=tel%3A%2B19585550100&
originatorName=Alice&
receiverAddress=tel%3A%2B19585550101&
receiverName=Bob&
mediaType=Video&
transportProtocol=RTP%2FAVP&
encodingName=H263-2000&
clockRate=90000&
fmtp=profile%3D0%3B%20level%3D45&
encodingName=MP4V-ES&
bandwidth=54&
bwType=AS&
frameRate=8&
```

```
clientCorrelator=12345
```

C.2.3.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:videoShareSessionInformation  xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
<originatorAddress>tel:+19585550100</originatorAddress>
 <originatorName>Alice</originatorName >
 <receiverAddress>tel:+19585550101</receiverAddress>
 <receiverName>Bob</receiverName>
<mediaInformation>
   <mediaType>Video</mediaType >
   <transportProtocol>RTP/AVP</transportProtocol>
   <mediaFormats>
     <fmt>96</fmt>
     <encodingName>H263-2000</encodingName>
     <clockRate>90000</clockRate>
     <fmtp>profile=0; level=45</fmtp>
   </mediaFormats>
   <mediaFormats>
     <fmt>97</fmt>
     <encodingName>MP4V-ES</encodingName>
   </mediaFormats>
   <bandwidth>54</bandwidth>
   <bwType>AS</bwType>
   <frameRate>8</frameRate> </mediaInformation>
<status>Initial</status>
<clientCorrelator>12345</clientCorrelator>
<resourceURL>http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</vs:videoShareSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section C.1.2.2).

C.3 Accepting a 1-1 video share session invitation

This operation is used for accepting a 1-1 video share invitation, by means of updating the status, see section 6.5.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
Status	SessionStatus	No	Status of the Receiver. To indicate that the user accepts the session invitation, this element MUST be set to “Connected”

mediaType	MediaType [0…2]	Yes	Type of the media a (e.g. “Audio” or “Video”). Maximum one entry for video media type and one entry for audio media type are allowed. See description of “m=” field in [RFC4566].
transportProtocol	xsd:string [0…2]	Yes	The transport protocol of the media (e.g. “RTP/AVP”, “RTP/SAVP”,”udp”). See description of “m=” field in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
fmt	xsd:string [0…2]	Yes	When “transportProtocol” is "RTP/AVP" or "RTP/SAVP”, it contains static or dynamic RTP payload type number. When static RTP payload type number is used, it SHALL be present in the request bodies during the session resource creation and the rest of the elements in this structure are not needed. The dynamic RTP payload type number is generated either by the server or by the client. If generated by the client, it SHALL be present in the request bodies during the session resource creation; if generated by the server, it SHALL not be present in the request bodies during the session resource creation. When dynamic RTP payload type number is used, at least “encodingName” SHALL be present. See description of “m=” field in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
encodingName	xsd:string [0…2]	Yes	The encoding name of the RTP payload type. See description of “a=rtpmap” field in [RFC4566]. It SHALL use media subtype (e.g., “AMR” for audio, “H263-2000” for video.) when “RTP/AVP" or "RTP/SAVP" transport protocol is used, see IANA for registered media subtypes for audio and video. Maximum one entry for video media type and one entry for audio media type are allowed.
clockRate	xsd:unsignedLong [0…2]	Yes	Number of samples per second, see description of “a=rtpmap” field in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
encodingParameter	xsd:string [0…unbounded]	Yes	List of encoding parameters for the RTP payload type. See description of “a=rtpmap” field in [RFC4566].

fmtp	xsd:string [0…2]	Yes	Format specific parameters of “a=fmtp” as defined in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
bandwidth	xsd: unsignedLong [0…2]	Yes	Band width in kilobits per second. See description of “b=” field in [RFC4566].
bwType	xsd:string [0…2]	Yes	Currently only CT" (Conference Total) and "AS" (Application Specific) are supported. See description of “b=” field in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
frameRate	xsd: decimal [0…2]	Yes	The maximum video frame rate in frames/sec, defined only for video media. See description of “a=framerate” field in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
pTime	xsd: unsignedInt [0…2]	Yes	The length of time in milliseconds represented by the media in a packet. It is probably only meaningful for audio data, but may be used with other media types if it makes sense. See description of “a=ptime” field in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
maxPTime	xsd: unsignedInt [0…2]	Yes	This gives the maximum amount of media that can be encapsulated in each packet, expressed as time in milliseconds. For frame-based codecs, the time SHOULD be an integer multiple of the frame size. It is probably only meaningful for audio data, but may be used with other media types if it makes sense. See description of “a=maxptime” field in [RFC4566]. Maximum one entry for video media type and one entry for audio media type are allowed.
attributeName	xsd:string [[0…unbounded]	Yes	Name of the attribute
attributeValue	xsd:string [0…unbounded]	Choice	The value of the attribute
<any element>	< type is defined in a schema implementing the element>	Choice	The value of the attribute. Note that element ‘any element’ can be any element from any other namespace (schema) than the target namespace. Type of such element is defined by the schema implementing the element.

	[0…unbounded]		In XML implementations, element “any” must be qualified with the namespace prefix.

C.3.1 Example1: Accepting a 1-1 video share invitation with accepted media information (Informative)

C.3.1.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

status=Connected&
mediaType=Video&
transportProtocol=RTP%2FAVP&
fmt=96&
encodingName=H263-2000&
clockRate=90000&
fmtp=profile%3D0%3B%20level%3D45&
bandwidth=54&
bwType=AS&
frameRate=8&
```

C.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<vs:receiverSessionStatusResp xmlns:vs ="urn:oma:xml:rest:netapi:videoshare:1">
  <mediaURL>http://example.com/received/userId/tel%3A%2B19585550101/20110728175159/file1</mediaURL>
</vs:receiverSessionStatusResp>
```

C.3.2 Example2: Accepting a 1-1 video share invitation without accepted media information (Informative)

C.3.2.1 Request

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

status=Connected
```

C.3.2.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

Appendix D. JSON examples (Informative)

JSON (JavaScript Object Notation) is a lightweight, text-based, language-independent data interchange format. It provides a simple means to represent basic name-value pairs, arrays and objects. JSON is relatively trivial to parse and evaluate using standard JavaScript libraries, and hence is suited for REST invocations from browsers or other processors with JavaScript engines. Further information on JSON can be found at [RFC4627].

The following examples show the request and response for various operations using the JSON data format. The examples follow the XML to JSON serialization rules in [REST_NetAPI_Common]. A JSON response can be obtained by using the content type negotiation mechanism specified in [REST_NetAPI_Common].

For full details on the operations themselves please refer to the section number indicated.

D.1 Reading all active video share notification subscriptions (section 6.1.3.1)

Request:

```
GET /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"videoShareSubscriptionList": {
   "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions",
   "videoShareNotificationSubscription": {
      "callbackReference": {
         "callbackData": "abcd",
         "notifyURL": "http://application.example.com/videoshare/notifications/77777"
      },
      "clientCorrelator": "12345",
      "duration": "7200",
      "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001"
   }
}}
```

D.2 Creating a new subscription to video share notifications using tel URI (section 6.1.5.1)

Request:

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/ HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"videoShareNotificationSubscription": {
```

```
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/videoshare/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200"
}}
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnnLocation: "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001
Date: Thu, 28 Jul 2011 17:51:59 GMT


{"videoShareNotificationSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/videoshare/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200",
   "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001"
}}
```

D.3 Creating a new subscription to video share notifications using ACR (section 6.1.5.2)

Request:

```
POST /exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/ HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"videoShareNotificationSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/videoshare/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/sub001
Date: Thu, 28 Jul 2011 17:51:59 GMT
```

{"resourceReference": {"resourceURL": "http://example.com/exampleAPI/videoshare/v1/acr%3Apseudonym123/subscriptions/sub001"}}

D.4 Reading an individual subscription (section 6.2.3.1)

Request:

```
GET /exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001?resFormat=JSON HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"videoShareNotificationSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/videoshare/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200",
   "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001"
}}
```

D.5 Cancelling a subscription (section 6.2.6.1)

Request:

```
DELETE /exampleAPI/videoshare/v1/tel%3A%2B19585550100/videoshare/subscriptions/sub001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jun 2010 17:51:59 GMT
```

D.6 Creating a new 1-1 video share session with mediaURL for recorded video (no CS call related) (section 6.3.5.1)

Request:

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com
```

```
{"videoShareSessionInformation": {
   "clientCorrelator": "12345",
   "mediaURL": "http://myvideos.com/recorded/holidays/20110501/file1/",
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"videoShareSessionInformation": {
   "clientCorrelator": "12345",
   "mediaURL": "http://myvideos.com/recorded/holidays/20110501/file1/",
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob",
   "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001",
   "status": "Initial"
}}
```

D.7 Creating a new 1-1 video share session with recorded video file content (CS call related) (section 6.3.5.2)

Request:

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
Accept: application/json
Host: example.com
MIME-Version: 1.0

--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"videoShareSessionInformation": {
   "callObjectRef": "http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001",
   "clientCorrelator": "12345",
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice"
}}
```

```
--===============123456==
Content-Disposition: form-data; name=" attachments ";filename="file1"
Content-Type: video/H263-2000
Content-Transfer-Encoding: binary
Content-Length: [length of video file]


 ...binary video file...

--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"videoShareSessionInformation": {
   "callObjectRef": "http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001",
   "clientCorrelator": "12345",
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob",
   "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001",
   "status": "Initial"
}}
```

D.8 Creating a new 1-1 video share session with live video (no CS call related) (section 6.3.5.3)

Request:

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"videoShareSessionInformation": {
   "clientCorrelator": "12345",
   "mediaInformation": {
      "bandwidth": "54",
      "bwType": "AS",
      "frameRate": "8",
      "mediaFormats": [
         {
            "clockRate": "90000",
            "encodingName": "H263-2000",
            "fmtp": "profile=0; level=45"
         },
```

```
            {"encodingName": "MP4V-ES"}
        ],
        "mediaType": "Video",
        "transportProtocol": "RTP/AVP"
    },
    "originatorAddress": "tel:+19585550100",
    "originatorName": "Alice",
    "receiverAddress": "tel:+19585550101",
    "receiverName": "Bob"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"videoShareSessionInformation": {
    "clientCorrelator": "12345",
    "mediaInformation": {
        "bandwidth": "54",
        "bwType": "AS",
        "frameRate": "8",
        "mediaFormats": [
            {
              “fmt”:”96”,
               "clockRate": "90000",
                "encodingName": "H263-2000",
                "fmtp": "profile=0; level=45"
            },
            {
              “fmt”:”97”,
              "encodingName": "MP4V-ES"
              }
        ],
        "mediaType": "Video",
        "transportProtocol": "RTP/AVP"
    },
    "originatorAddress": "tel:+19585550100",
    "originatorName": "Alice",
    "receiverAddress": "tel:+19585550101",
    "receiverName": "Bob",
    "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001",
    "status": "Initial"
}}
```

D.9 Retrieving 1-1 video share session information (section 6.4.3.1)

Request:

```
GET /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"videoShareSessionInformation": {
   "callObjectRef": "http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001",
   "clientCorrelator": "12345",
   "mediaURL": "http://myvideos.com/recorded/holidays/20110501/file1/",
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob",
   "resourceURL": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001",
   "status": "Connected"
}}
```

D.10 Terminating a 1-1 video share session (section 6.4.6.1)

Request:

```
DELETE /exampleAPI/videoshare/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.11 Accepting a 1-1 video share invitation with accepted media information (section 6.5.5.1)

Request:

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"receiverSessionStatus": {
   "mediaInformation": {
      "bandwidth": "54",
      "bwType": "AS",
      "frameRate": "8",
```

```
        "mediaFormats": {
            "clockRate": "90000",
            "encodingName": "H263-2000",
            "fmt": "96",
            "fmtp": "profile=0; level=45"
        },
        "mediaType": "Video",
        "transportProtocol": "RTP/AVP"
    },
    "status": "Connected"
}}
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"receiverSessionStatusResp": {"mediaURL": "http://example.com/received/userId/tel%3A%2B19585550101/20110728175159/file1"}}
```

D.12 Accepting a 1-1 video share invitation without accepted media information (section 6.5.5.2)

Request:

```
POST /exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"receiverSessionStatus": {"status": "Connected"}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.13 Notify a client about 1-1 video share session invitations (no CS call related) (section 6.6.5.1)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com
```

```
{"sessionInvitationNotification": {
   "link": [
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
         "rel": "VideoShareSessionInformation"
      },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status",
         "rel": "ReceiverSessionstatus"
      },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": "VideoShareNotificationSubscription"
      }
   ],
   "mediaInformation": {
      "bandwidth": "54",
      "bwType": "AS",
      "frameRate": "8",
      "mediaFormats": [
         {
            "clockRate": "90000",
            "encodingName": "H263-2000",
            "fmt": "96",
            "fmtp": "profile=0; level=45"
         },
         {
            "encodingName": "MP4V-ES",
            "fmt": "97"
         }
      ],
      "mediaType": "Video",
      "transportProtocol": "RTP/AVP"
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob"
}}
```


Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.14 Notify a client about 1-1 video share session invitations (CS call related) (section 6.6.5.2)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
```

```
Content-Length: nnnn
Accept: application/json
Host: application.example.com

{"sessionInvitationNotification": {
   "callObjectRef": "http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001",
   "link": [
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
         "rel": "VideoShareSessionInformation"
      },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001/status",
         "rel": "ReceiverSessionstatus"
      },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": "VideoShareNotificationSubscription"
      }
   ],
   "mediaInformation": {
      "bandwidth": "54",
      "bwType": "AS",
      "frameRate": "8",
      "mediaFormats": [
         {
            "clockRate": "90000",
            "encodingName": "H263-2000",
            "fmt": "96",
            "fmtp": "profile=0; level=45"
         },
         {
            "encodingName": "MP4V-ES",
            "fmt": "97"
         }
      ],
      "mediaType": "Video",
      "transportProtocol": "RTP/AVP"
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.15 Notify a client about the acceptance of 1-1 video share session with recorded video (no CS call related) (section 6.7.5.1)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com

{"sessionAcceptanceNotification": {
   "link": [
   {
      "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
      "rel": "VideoShareSessionInformation"
   },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": " VideoShareNotificationSubscription "
   }
   ]
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob",
   "status": "Connected"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.16 Notify a client about the acceptance of 1-1 video share session with recorded video (CS call related) (section 6.7.5.2)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com


{"sessionAcceptanceNotification": {
   "callObjectRef": "http://example.com/exampleAPI/call/tel%3A%2B19585550101/sessions/callSess001",
```

```
   "link": [
   {
      "href": "http://example.com/exampleAPI/videoshare/v1 tel%3A%2B19585550101/sessions/sess001",
      "rel": "VideoShareSessionInformation"
   },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": " VideoShareNotificationSubscription "
      }
   ],
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob",
   "status": "Connected"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.17 Notify a client about the acceptance of 1-1 video share session with live video (no CS call related) (section 6.7.5.3)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com


{"sessionAcceptanceNotification": {
   "link": [
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
         "rel": "VideoShareSessionInformation"
      },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": "VideoShareNotificationSubscription"
      }
   ],
   "mediaInformation": {
      "bandwidth": "54",
      "bwType": "AS",
      "frameRate": "8",
      "mediaFormats": {
         "clockRate": "90000",
         "encodingName": "H263-2000",
         "fmt": "96",
         "fmtp": "profile=0; level=45"
      },
```

```
      "mediaType": "Video",
      "transportProtocol": "RTP/AVP"
   },
   "mediaURL": "http://application.example.com/userId/tel%3A%2B19585550101/20110728175159/file1",
   "receiverAddress": "tel:+19585550101",
   "receiverName": "Bob",
   "status": "Connected"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.18 Notify a client about 1-1 video share event (ended) (section 6.8.5.1)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com

{"videoShareEventNotification": {
   "eventType": "SessionEnded",
   "link": [
   {
      "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
      "rel": "VideoShareSessionInformation"
   },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": " VideoShareNotificationSubscription "
      }
   ]
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.19 Notify a client about 1-1 video share event (declined) (section 6.8.5.2)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
```

```
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com

{"videoShareEventNotification": {
   "eventType": "Declined",
   "link": [
   {
      "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
      "rel": "VideoShareSessionInformation"
   },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": " VideoShareNotificationSubscription "
      }
   ]
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.20 Notify a client about 1-1 video share event (cancelled) (section 6.8.5.3)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com

{"videoShareEventNotification": {
   "eventType": "SessionCancelled",
   "link": [
   {
      "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
      "rel": "VideoShareSessionInformation"
   },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": " VideoShareNotificationSubscription "
      }
   ]
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.21 Notify a client about video share event (failed) (section 6.8.5.4)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com

{"videoShareEventNotification": {
   "eventType": "Failed",
   "link": [
   {
      "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/sessions/sess001",
      "rel": "VideoShareSessionInformation"
   },
      {
         "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550101/subscriptions/sub001",
         "rel": " VideoShareNotificationSubscription "
      }
   ]
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.22 Notify a client about subscription cancellation (section 6.9.5.1)

Request:

```
POST /videoshare/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: application.example.com

{"subscriptionCancellationNotification": {
   "callbackData": "abcd",
   "link": {
      "href": "http://example.com/exampleAPI/videoshare/v1/tel%3A%2B19585550100/subscriptions/sub001",
      "rel": "VideoShareNotificationSubscription"
   }
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

Appendix E. Operations mapping to a pre-existing baseline specification (Informative)

As this specification does not have a baseline specification, this appendix is empty.

Appendix F. Light-weight resources (Informative)

As this version of the specification does not define any light-weight resources, this Appendix is empty.

Appendix G. Authorization aspects (Normative)

This appendix specifies how to use the RESTful Video Share API in combination with some authorization frameworks.

G.1 Use of Autho4API

The RESTful Video Share API MAY support the Autho4API authorization framework defined in [Autho4API_10].

A RESTful Video Share API supporting [Autho4API_10]:

- SHALL conform to section D.1 of [REST_NetAPI_Common];
- SHALL conform to this section G.1.

G.1.1 Scope values

G.1.1.1 Definitions

In compliance with [Autho4API_10], an authorization server serving clients requests for getting authorized access to the resources exposed by the RESTful Video Share API:

- SHALL support the scope values defined in Table below;
- MAY support scope values not defined in this specification.

Scope value	Description	For one-time access token
oma_rest_videoshare.all_{apiVersion}	Provide access to all defined operations on the resources in this version of the API. The {apiVersion} part of this identifier SHALL have the same value as the “apiVersion” URL variable which is defined in section 5.1. This scope value is the union of the other scope values listed in next rows of this table.	No
oma_rest_videoshare.sessions	Provide access to all defined operations on 1-1 video share sessions	No
oma_rest_videoshare.subscr	Provide access to all defined operations on video share subscriptions	No

Table 5: Autho4API scope values for RESTful Video Share API

G.1.1.2 Downscoping

In the case where the Autho4API client requests authorization for “ oma_rest_videoshare.all_{apiVersion}” scope, the Autho4API Authorization Server and/or resource owner MAY restrict the granted scope to some of the following scope values:

- “oma_rest_videoshare.sessions”
- “oma_rest_videoshare.subscr”

G.1.1.3 Mapping with resources and methods

Tables in this section specify how the scope values defined in section G.1.1.1 for the RESTful Video Share API map to the REST resources and methods of this API. In these tables, the root “oma_rest_videoshare.” of scope values is omitted for readability reasons.

Resource	URL Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
All 1-1 video share sessions	/sessions	6.3	n/a	n/a	**all_{apiVersion} or sessions**	n/a
Individual 1-1 video share session	/sessions/{sessionId}	6.4	**all_{apiVersion} or sessions**	n/a	n/a	**all_{apiVersion} or sessions**
Individual 1-1 video share session status	/sessions/{sessionId}/status	6.5	n/a	n/a	**all_{apiVersion} or sessions**	n/a

Table 6: Required scope values for: 1-1 video share sessions

Resource	URL Base URL: http://{serverRoot}/videoshare/{apiVersion}/{userId}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions to video share notifications	/subscriptions	6.1	**all_{apiVersion} or subscr**	n/a	**all_{apiVersion} or subscr**	n/a
Individual subscription to video share notifications	/subscriptions/{subscriptionId}	6.2	**all_{apiVersion} or subscr**	n/a	n/a	**all_{apiVersion} or subscr**

Table 7: Required scope values for: video share subscriptions

G.1.2 Use of 'acr:Authorization'

This section specifies the use of 'acr:Authorization' in place of an end user identifier in a resource URL path.

An 'acr' URI of the form 'acr:Authorization', where 'Authorization' is a reserved keyword MAY be used to avoid exposing a real end user identifier in the resource URL path.

A client MAY use 'acr:Authorization' in a resource URL in place of the {userId} resource URL variable in the resource URL path, when the RESTful Video Share API is used in combination with [Autho4API_10].

In the case the RESTful Video Share API supports [Autho4API_10], the server:

- SHALL accept 'acr:Authorization' as a valid value for the resource URL variable {endUserId}.
- SHALL conform to [REST_Common_TS] section 5.8.1.1 regarding the processing of 'acr:Authorization'.