VOD Interface Part 5d

STB - TRAXIS.SRM - VideoServer

PROPRIETARY NOTICE

This document is the property of SeaChange International. All information herein is confidential and commercially sensitive to SeaChange International and must not be copied or disclosed to any third party without the prior written consent of SeaChange International.

Disclosure is permissible provided always that such a disclosure shall only be to people who are directly involved in activities to which this document relates and who have a “need to know”

LEGAL DISCLAIMER

This specification is provided "as is" and without any warranty or representation of any kind, express or implied. Without limitation, there is no warranty of non-infringement, no warranty of merchantability, and no warranty of fitness for a particular purpose. All warranties are expressly disclaimed. User assumes the full risk of using the specification. In no event shall SeaChange International be liable for any actual, direct, indirect, punitive, or consequential damages arising from such use, even if advised of the possibility of such damages.

History

Version	Status	Date	Author	Description
0.100	Proposal	20/04/10	W.R. Dittmer	Added example error for session notification
0.101	Proposal	16/08/10	W.R. Dittmer	Added EMM header description on SETUP response.
1.0	Proposal	26/10/10	J.Verhoeven/R.Kochen	Added Trick history specification. History 0.1 – 0.99 Deleted.
1.0	Released	12-11-2010	Bert van Willigen	Doc status changed to released.
1.01	Released	26/07/2011	W. van Boesschoten	Minor editorial fixes
1.1	Released	07/08/2012	SeaChange	Added AV-source to transport header in RTSP SETUP response for IPTV clients → to allow for “NAT hole-puncing” by IPTV client if behind NAT.
1.11	Released	08/08/2012	SeaChange	Fix on addition done in version 1.1
1.2	Proposal	01/10/2012	M. Cuijpers	Clarifications on trick-play mask being decimal, date/time formatting and session history (sent on teardown).
1.3	Proposed	23/11/2012	R. Janssen/K. Seiler	Added Annuonce Code 2401 Extended description for resuming bookmark in case of playlist Document branding
1.4	Proposed	10/01/2012	E. Riemens	Added optional CADeviceId to identify a Home Gateway Device to the SETUP request

Contents

1 Introduction

1.1 Purpose of the document

Video On Demand (VOD) systems contain many internal- and external systems and subsystems that have to be linked to each other. The purpose of this document is to describe the interfaces between those systems for the VOD head-end architecture.

1.2 Scope

This document deals only with the interfaces related to the VOD infrastructure. It does not describe any head-end components that are involved in the delivery of the 'unaddressed broadcasting services' to the customer.

1.3 Changes

The interfaces in this document are specified as accurately as possible at the time of the writing. SeaChange keeps the right to improve the interfaces. Suppliers using the interfaces will be notified of such changes.

1.4 Acronyms, Abbreviations and Terms

The next list provides an overview of acronyms and abbreviations used in this document and where they stand for.

Acronym		Stands for
ADI		Asset Distribution Interface
CMS		Content Management System
CRM		Customer Relationship Management
DPI		Dots Per Inch
DVB		Digital Video Broadcasting
EPG		Electronic Program Guide
FTP		File Transfer Protocol
FVOD		Free Video On Demand
HD		High Definition
HTTP		HyperText Transfer Protocol
ICC		International Color Consortium
ID		IDentity
ISO		International Standards Organization
JP(E)G		Joint Photographic (Expert) Group
LDAP		Lightweight Directory Access Protocol
MAC		Media Access Control
MD5		Message Digest Algorithm 5
MPEG		Moving Picture Experts Group
PNG		Portable Network Graphics
RTSP		Real Time Streaming Protocol
SOAP		Simple Object Access Protocol
SPTS		Single Program Transport Stream
STB		Set Top Box
SVOD		Subscription Video On Demand
TAR		Tape ARchive*
TVOD		Transaction Video On Demand

UBF		United Broadcast Facilities
USTAR		Uniform Standard Tape ARchive
UTC		Coordinated Universal Time
VOD		Video on demand
XML		eXtensible Markup Language

Below terms are listed with their definitions, as used in this document.

Term	**Definition**
Asset	Content together with metadata.
Asset group	A bundle of related assets, e.g. the feature film, its trailer and the ‘making of…’. The asset group has nothing to do with a commercial offering (products).
Banner	A headline containing image- and/or text essences
Content	A bundle of related essences, e.g., an SPTS with an audio and a video stream.
Content Provider	The original provider of the content
Content Packager	The facilitator contracted by the MSO to do the content packaging and propagation of the content to the MSO’s system
Essence	The data that represents pictures, sound and text; types of essence include video, audio and data of various kinds, including captions, graphics still images, text enhancements and other data as needed by each application
File drop	The process of putting a (content-)file in a dedicated part (folder) of a server
License window	The time period in which the MSO is allowed to show to its customers
Metadata	The generic term for all sorts of captured data that relates in one way or another to essence material; media related metadata (SD/HD, 4:3 vs 16:9, etc), title metadata (title, description, directors etc) and terms metadata (e.g. rental period).
Poster	Image asset to be shown in the catalogue.
Preview/trailer	Asset which is the item that will (often) be delivered free of charge as an advance viewing of several scenes advertising a forthcoming movie.
PRODIS	An SeaChange system supporting the management of assets and products
Product	A commercial offer, to be sold to consumers, consisting of one or more assets
Title	An identifying name given to an asset
TRAXIS	An SeaChange system supporting processes involved in VOD transactions
Rental period / Viewing period	The time period following the actual product purchase by the consumer for which the product may be viewed before a new purchase would be required.

1.5 Conventions

The following conventions are applicable in this document:

- The word *shall* is used to indicate mandatory requirements strictly to be followed and from which no deviation is permitted *(shall* equals *is required to)*.
- The word *should* is used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited *(should* equals *is recommended that)*.
- The word *must* is used only to describe unavoidable situations.
- The word *will* is only used in statements of fact.
- The word *may* is used to indicate a course of action permissible within certain limits *(may* equals *is permitted to)*.
- The word *can* is used for statements of possibility and capability, whether material, physical, or causal *(can* equals *is able to)*.

1.6 Standards

Date time formats between the systems are ALWAYS in UTC time and W3C (ISO 8601 profile) formatting, e.g.: *2004-11-05T13:15:30Z*. This way time discontinuities can be avoided (daylight savings). Note that all interfacing systems must decode/encode the date time to the correct local time.

1.7 References

This chapter lists the references made in this document.

[RFC_2326]	Real Time Streaming Protocol (RTSP), IETF RFC 2326
[EN_300468]	ETSI EN 300 468 v1.7.1 (2006-05) Digital Video Broadcast; Specification for Service Information (SI) in DVB systems
[TOP_XSD]	Topology.xsd
[TOP_RM]	Topology Excel sheet readme (TopologyExcelReadme.docx)
[OTV_RTSP]	OpenTV RTSP Interoperability Guide Q1_2008 – Version 5.3

1.8 Open Issues

2 STB → TRAXIS.SRM / Streaming server

This section defines the interface between the STB on-demand client, the SRM (i.e. TRAXIS.SRM) and the Streaming server for the setup and teardown of streaming sessions at the Streaming server. The message protocol (and mechanisms/semantics) is based on the RTSP protocol.

An end-to-end streaming session involves the following main components:

- STB on-demand client → initiates the setup and the teardown of the “end-to-end” streaming session (on request of the end-user)
- TRAXIS.SRM provides the Session & Resource Management → based on the location of the requesting STB in the on-demand network:
  - In a DVB-C deployment TRAXIS.SRM determines which Streaming server is able to reach the Edge-QAM which on its turn is able to deliver the requested stream at the location of the STB.
  - In an IPTV deployment TRAXIS.SRM determines which Streaming server is able to stream to the requesting STB.
- Streaming server, managed by the TRAXIS.SRM → streams, the by the end-user requested asset, to the STB (in a DVB-C deployment this is “via” the Edge-QAM).

2.1 Session setup

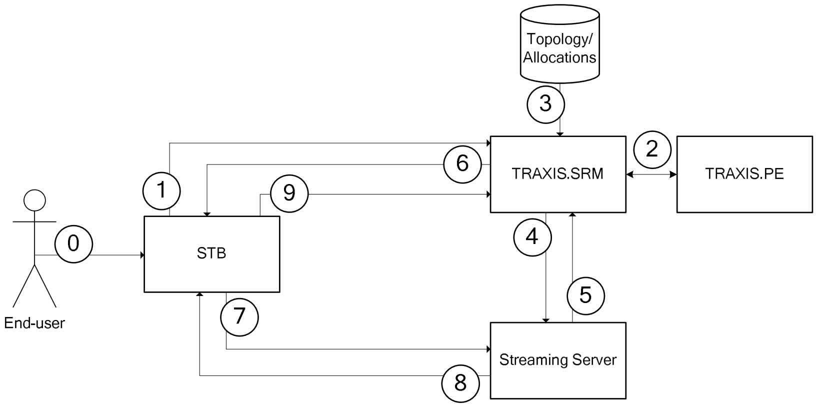

FIGURE 1: STREAMING SESSION SETUP

The figure above depicts the message exchange during a streaming session setup.

- The end-user initiates the setup of the streaming session for a particular asset via the on-demand-client (0)
- The on-demand client addresses the session manager (i.e. TRAXIS.SRM) with the request to setup a streaming session (1) → RTSP SETUP @ SRM
- TRAXIS.SRM optionally authorizes the request to setup a streaming session for the requested asset with TRAXIS.PE (2). TRAXIS.PE is the Purchase & Entitlement Manager and maintains the on-demand entitlements per end-user [TRAXIS_PE].
- Based on the serving area of the requesting STB, TRAXIS.SRM determines for which Streaming server and Edge-QAM to allocate a stream (3).
  - The Edge-QAM allocation is only applicable in a DVB-C deployment.
- TRAXIS.SRM instructs the allocated Streaming server to setup a stream
  - In a DVB-C deployment → to the allocated Edge-QAM (4)
  - In an IPTV deployment → to the requesting STB (4)

And the Streaming server returns the session identification (5) → RTSP SETUP @ Streaming server.

- Upon a successful session setup on the Streaming server, TRAXIS.SRM returns towards the STB :
    - the session identification of the session setup by the STB on the TRAXIS.SRM
    - the session identification of the session setup by TRAXIS.SRM at the Streaming server
    - the location (URI) of the streaming session control port on the Streaming Server
- The on-demand client issues a start streaming command at the Streaming server (7) and the Streaming server responds appropriately (8) → RTSP PLAY @ Streaming server
- Once the streaming has started the on-demand client is expected to send at regular intervals a keep-alive message (9). In the absence of the keep-alive message TRAXIS.SRM may decide to tear down the streaming session → RTSP GET_PARAMETER @ SRM

2.2 Session teardown

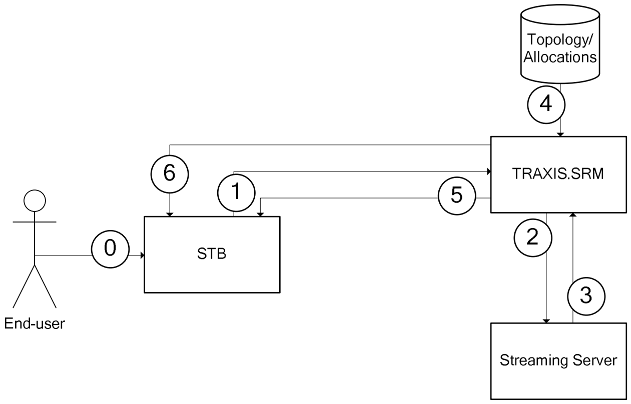

FIGURE 2: STREAMING SESSION TEARDOWN

The streaming session teardown message exchange includes:

- The end-user initiates the teardown of the session via the STB's on-demand client (0)
- The on-demand client addresses TRAXIS.SRM to tear down the session at the TRAXIS.SRM (1)
- TRAXIS.SRM instructs the Streaming server to tear down the corresponding session at the Streaming server (2). After the Streaming server torn down the session it shall respond with the confirmation of the session teardown (3)
- TRAXIS.SRM releases the Streaming server resources and in a DVB-C deployment also the Edge-QAM resources allocated to the session (4) and inform the on-demand client that the session has been torn-down (5).

Note that, upon a keep-alive time-out, a streaming session teardown may also be initiated by TRAXIS.SRM

- TRAXIS.SRM informs the on-demand client that the streaming session will be torn down (6). The on-demand client may respond to the announcement but TRAXIS.SRM does not expect back to receive a response.
- TRAXIS.SRM instructs the streaming server to tear down the session (2), (3) and releases the allocated session resources (4).

2.3 TCP-connections

2.3.1 STB – TRAXIS.SRM TCP-connection

In case TCP is used as the network protocol for the exchange of RTSP messages, each STB shall create a single persistent TCP-connection to TRAXIS.SRM at the time of the session setup initiation which shall be used for the streaming session control (i.e. setup and teardown requests).

The STB (i.e. the client) is responsible for the TCP-connection. In case of a connection failure it is the responsibility of the STB to re-establish the TCP-connection between the TRAXIS.SRM and the STB. TRAXIS.SRM shall take no explicit actions in case the TCP-connection is broken. Note that a broken TCP-connection implies that the STB is no longer able to issue the required keep-alive messages upon which TRAXIS.SRM may decide to terminate the session. However if the connection is re-established in time (i.e. before reaching a session time-out) TRAXIS.SRM shall not tear down the session.

Although TRAXIS.SRM may be deployed in a multi-server configuration, TRAXIS.SRM shall manifest itself as a single logical entity with a single virtual IP-address towards the STBs.

2.3.2 STB – Streaming server TCP-connection

After a successful session setup between the STB and TRAXIS.SRM, the STB shall establish a (persistent) TCP-connection with the Streaming server (the server as indicated in the RTSP SETUP response). The TCP-connection shall be used to send the streaming control messages (i.e. RTSP PLAY, RTSP PAUSE) towards the Streaming server.

2.4 UDP

2.4.1 STB – TRAXIS.SRM

In case UDP is used as the network protocol for the exchange of RTSP messages, it is not possible to maintain a persistent connection between the STB and TRAXIS.SRM. Therefore it is essential that the STB sends keep-alive messages at a regular interval, in the absence of the keep-alive messages TRAXIS.SRM shall tear down the session.

Due to the "unreliable" nature of UDP the STB is expected to resend the previous RTSP request if a response from the TRAXIS.SRM is not received within a time-out value after sending the request. The STB shall assume the request has been lost.

2.4.2 STB – Streaming server

After a successful session setup between the STB and TRAXIS.SRM via, the STB shall also use UDP for the delivery streaming control messages (i.e. RTSP PLAY, RTSP PAUSE) towards the Streaming server (the server as indicated in the RTSP SETUP response).

2.5 RTSP Session

An individual client session is identified by its Session identification as determined/defined by the RTSP-server upon the session setup request – this is independent of the underlying network protocol (i.e. TCP or UDP) , i.e.

- The (TCP-) connection does not need to persist during the lifetime of the session
- Multiple (TCP-) connections / UDP sources can be used to address an individual session
- Multiple sessions can be addressed via a single (TCP-) connection / UDP source.

In general, a RTSP-server knows 3 states per client session (discard of CAPTURING state)

- INIT → Idle state no SETUP received
- READY → SETUP request successfully handled (also after successful handling of PAUSE request)
- PLAYING → PLAY request successfully handled

TRAXIS.SRM will only know the INIT and READY states

- INIT → READY upon RTSP SETUP request from client
- READY → INIT upon RTSP TEARDOWN request from client

TRAXIS.SRM shall only handle the RTSP SETUP and TEARDOWN messages from the client – TRAXIS.SRM will not implement the RTSP PLAY and RTSP PAUSE messages. Upon reception of a PLAY or PAUSE request message TRAXIS.SRM will respond with the appropriate error response (i.e. Method not allowed).

Streaming server will know all 3 states

- INIT → READY upon RTSP SETUP from SRM
- READY → PLAYING upon RTSP PLAY from client
- PLAYING → READY upon RTSP PAUSE from client
- READY or PLAYING → INIT upon RTSP TEARDOWN from SRM

The Streaming server shall only handle the RTSP SETUP and RTSP TEARDOWN messages from the TRAXIS.SRM. RTSP SETUP and TEARDOWN request issued by the STB on-demand client shall not be honored, the Streaming server shall reply with the appropriate error response (i.e. Method not allowed).

2.5.1 Session setup

An individual on-demand client shall maintain 2 connections,

- Connection to TRAXIS.SRM for session control
- Connection to Streaming server for stream control

0. The on-demand client addresses the setup of the streaming session at the TRAXIS.SRM via a RTSP SETUP request, including (among others) the parameters:
   a. Asset
   b. Service-group of the on-demand client
1. Based on the Serving-group of the client, the TRAXIS.SRM allocates a Streaming-server stream and for DVB-C the Edge-QAM stream for the session. If no resources are available the TRAXIS.SRM will return the appropriate error response (i.e. Not enough bandwidth).
2. The TRAXIS.SRM sets up a session at the Streaming server by means of a RTSP SETUP request. The request must include:
   a. The identification of the asset
   b. The destination IP-address/port-number of the stream (i.e. in a DVB-C deployment the input at the Edge-QAM, in a IPTV deployment the streaming input of the requesting STB)
3. The Streaming server creates a Streaming session (state=READY) and returns the RTSP SETUP response to the TRAXIS.SRM, including:
   a. Session identification to be used by a client to identify this stream at the Streaming server (i.e. used by the on-demand client in the RTSP PLAY and RTSP PAUSE requests).
   b. The Streaming server's control RTSP URI to be used by the client to control the stream
4. The TRAXIS.SRM creates the streaming session and returns on its turn the RTSP SETUP response to the on-demand client including:
   a. Identification of session between TRAXIS.SRM and client
   b. RF Tuning parameters (in a DVB-C deployment) or IP stream destination (in an IPTV deployment)
   c. Identification of session between Streaming server and client

d. Stream-server control RTSP URI

2.5.2 Session control

For session control the on-demand client addresses the Streaming server directly via the stream control RTSP URI presented by the TRAXIS.SRM included in the RTSP SETUP response.

0. On-demand client creates a TCP-connection with the Streaming server (IP-address/FQDN and port as presented by the <host>:<port> of the RTSP URI)
1. The client sends a RTSP PLAY to the Streaming-server, after successfully handling the RTSP PLAY request, the state of the session is promoted to PLAYING
    a. Session identified by value as presented by the Streaming server upon the session setup by TRAXIS.SRM and returned to client by TRAXIS.SRM as part of the RTSP SETUP response
2. The client sends RTSP PAUSE, after successfully handling the request the state of the session will become READY
    a. Optionally, upon a “pause timeout” the Streaming server shall send a RTSP ANNOUNCE to the TRAXIS.SRM. TRAXIS.SRM may decide to tear down the session upon reception of the message.
3. The Streaming server sends (unsolicited) RTSP ANNOUNCE messages to the on-demand client to inform the client about the status of the streaming session (e.g. end-of-stream reached etc.)

2.5.3 Session teardown

The on-demand client shall address the RTSP TEARDOWN request at the TRAXIS.SRM, which on its turn will instruct the Streaming server to tear down the streaming session.

0. On-demand client sends a RTSP TEARDOWN request towards TRAXIS.SRM, where the session is identified by the value, provided at the session setup between on-demand client and TRAXIS.SRM.
1. TRAXIS.SRM instructs the Streaming server to teardown the session at the Streaming server, where the session is identified by the value provided at the session setup between TRAXIS.SRM and Streaming server.
2. Streaming server tears down the streaming session returns RTSP TEARDOWN response to TRAXIS.SRM
    a. The Streaming server closes the TCP-connection between the Streaming server and the STB for the corresponding session.
3. TRAXIS.SRM returns RTSP TEARDOWN response to the on-demand client
    a. TRAXIS.SRM closes the TCP-connection between the TRAXIS.SRM and the STB for the corresponding session.

2.5.4 Session time-out

The on-demand client is expected to send a keep-alive message to the SRM at a regular interval. If TRAXIS.SRM detects a session inactivity time-out, TRAXIS.SRM may:

1. If the TCP-connection between TRAXIS.SRM and STB is established, send a RTSP ANNOUNCE message to the client to inform the client that the session shall be torn-down (TRAXIS.SRM shall not send an ANNOUNCE message in case UDP is used as the underlying network protocol)
    a. Including the reason of the teardown
    b. TRAXIS.SRM closes the TCP-connection for the corresponding session
2. Instruct the Streaming server to tear down the session in question where the session is identified by the value provided at the session setup between TRAXIS.SRM and Streaming server.
    a. If established, the Streaming server closes the TCP-connection between the Streaming server and the STB for the corresponding session.

2.6 STB – TRAXIS.SRM TCP-connection timeout and recovery

The STB is expected to send at regular intervals a keep-alive message to inform the TRAXIS.SRM about its status. If the STB fails to send the next keep-alive message within the time-out period, TRAXIS.SRM may decide to close the connection after connectivity time-out – i.e. if TRAXIS.SRM did not receive a keep-alive message from the STB after a (configurable) time-out.

2.6.1 STB or on-demand client failure

0. STB failure
   a. The streaming session is unexpectedly terminated at the STB without informing TRAXIS.SRM
1. STB re-starts and the on-demand client sets up a new streaming session
2. TRAXIS.SRM detects that a “second streaming session” is initiated for the requesting STB and shall
   a. tear down the first streaming session at the Streaming server (i.e. the session setup before the STB failure occurred)
   b. and release the allocated resources.
3. TRAXIS.SRM handles the new streaming session setup as any session setup.

Note that if the STB fails to re-establish the connection before the session is cleared due to an inactivity time-out (i.e. STB failed to send the keep-alive message) TRAXIS.SRM may already cleared the session in which case set up of the new session follows the normal session setup flow.

2.6.2 TRAXIS.SRM recovery or TCP-connection recovery

The STB has no means to distinguish between a TRAXIS.SRM failure and a TCP-connection failure, in both cases the TCP-connection is lost.

0. TCP-connection between STB and TRAXIS.SRM is lost
1. STB tries to re-establish at regular intervals the TCP-connection (until TRAXIS.SRM or TCP-connection is recovered)
2. TRAXIS.SRM accepts re-established TCP-connection
3. STB continues its message exchange with TRAXIS.SRM
   a. TRAXIS.SRM persists all active sessions, therefore there is no need for the STB to “re-setup” the session

Note that the above is only valid in case a TCP-connection is used for the delivery of the RTSP messages. In case UDP is used TRAXIS.SRM shall simply not reply with a response on the RTSP requests in case of a TRAXIS.SRM failure. The STB is expected to retry the request until TRAXIS.SRM is recovered from its failure, i.e. the STB should treat this as any other timed-out request.

2.7 STB – TRAXIS.SRM RTSP message syntax

The RTSP messages follow in syntax, semantics and operations RFC2326.

The RTSP message requests and responses signatures described below presents the SeaChange’ default RTSP SETUP and TEARDOWN implementation. However additional support for other RTSP client implementations is provided; refer to the appropriate specification for details.

User-agent	RTSP Client implementation
`eventIS_VOD_V1`	SeaChange RTSP Specification
`OpenTV VOD 1`	OpenTV VOD RTSP Specification [OTV_RTSP]

2.7.1 RTSP SETUP Request

```
SETUP rtsp://<host>:<port>/<asset>?VODServingAreaId=<VODServingAreaId>
&STBId=<STBId>[&CADeviceId=<CADeviceId>][&Entitlement=<Entitlement>][1] RTSP/1.0
```

Where:

- VODServingAreaId: The identification of the STB's serving area
- STBId: The identification of the requesting STB
- CADeviceId: The identification of the CA Device, e.g. a Home Gateway STB, through which the request is routed. (optional). When present, TRAXIS will use this identity to authenticate the device and/or to verify device authorization to access the content.
- Entitlement: The entitlement acquired during the purchase process (optional)

If TRAXIS is also used as the Purchase & Entitlement server (i.e. TRAXIS.PE) TRAXIS shall validate whether the STB is authorized to setup a streaming session for the requested asset.

Header	Opt.	Usage
CSeq	No	Number
User-Agent	No	On-demand client / SW-stack identification according. For the default SeaChange implementation this is: eventIS_VOD_V1
For DVB-C only: Transport	No	**For user-agent: eventIS_VOD_V1:**
		MP2T/DVBC/QAM;unicast
		For user-agent: OpenTV VOD 1:
		MP2T/DVBC/QAM;unicast;client=<client> Where <client> is the unique identification of the client device e.g. STB
For IPTV only: Transport	No	**For user-agent: eventIS_VOD_V1:**
		MP2T/AVP/UDP;unicast;destination=<destination>; client_port=<client_port> Where <destination> and <client_port> is the destination IP-address, port number of the stream
		For user-agent: OpenTV VOD 1:
		MP2T/**AVP**/**UDP**;unicast;client=<client> Where <client> is the unique identification of the client device e.g. STB
BillingID	Yes	**For user-agent: OpenTV VOD 1:**
		Deployment dependent client related billing information passed from client to server.

[1] The Entitlement=<Entitlement> query parameter may be added to preserve backwards compatibility with the previous version of the interface.

2.7.2 RTSP SETUP Response

`RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ TRAXIS.SRM `[;timeout=`<timeout>`]` The client is expected to send a keep-alive message within 80% of the time-out value.
`ControlSession`	No	Session identification @ Streaming server
For DVB-C only: `Tuning`	No	**For user-agent: `eventIS_VOD_V1`:** `frequency=`<frequency> `[;symbol_rate=`<symbol_rate>`]` **[**`;modulation=`<modulation>`]` `[;fec_inner=`<fec_inner>`]` `[;fec_outer=`<fec_outer>`]` (all values except frequency are optional) Where all values adhere to the DVB cable delivery system descriptor according [EN_300468]. **For user-agent: `OpenTV VOD 1`:** `frequency=`<frequency>`;symbol_rate=`<symbol_rate>`;modulation=`<modulation>`;fec_inner=`<fec_inner>`;fec_outer=`<fec_outer>`]` Where all values adhere to the DVB cable delivery system descriptor according [EN_300468].
For DVB-C only: `Channel`	No	`Tsid=`<Tsid>`;Svcid=`<Svcid> Where <Tsid> denotes the on-demand transport stream identification and <Svcid> the program number (service) in the transport stream
For IPTV only: `Transport`	No	`MP2T/DVBC/UDP;unicast;destination=`<destination>;`client_port=`<client_port>;`source=`<source>;`server_port=`<server_port> Where • <destination> coincides with the <destination> in the transport header of the RTSP SETUP request. • <source> and <server_port> is the source IP-address and port number of the stream
`Location`	No	Stream control RTSP URI
`Date`	Yes	Timestamp of the response
`EMM`	Yes	A string representation of the EMM. For Irdeto and Conax this will be a base64 encoded string of the EMM bytes.

Message body	Opt.	Usage

Position	Yes	Current stream position in NPT Position: <position>
Duration	Yes	Total duration of the asset (or playlist) Duration: <duration>

2.7.3 RTSP TEARDOWN Request

TEARDOWN * RTSP/1.0 - the TEARDOWN command uses the '*' character for URI to match the current presentation.

Header	Opt.	Usage
CSeq	No	Number
Session	No	Session identification @ TRAXIS.SRM
Reason	Yes	Reason for the teardown – can be used for logging purposes

2.7.4 RTSP TEARDOWN Response

RTSP/1.0 200 OK - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
CSeq	No	Number
Date	Yes	Timestamp of the response

2.7.5 RTSP GET_PARAMETER Request

GET_PARAMETER * RTSP/1.0 – the GET_PARAMETER command uses the '*' character for the URI, to match the current presentation.

Header	Opt.	Usage
CSeq	No	Number
Session	No	Session identification @ TRAXIS.SRM
Accept/ Content-Type *)	No	text/parameters
Content-Length	No	Length of RTSP request body body (= 0 if message body of request == <<empty>>)

Message body	Usage
<<empty>>	Used as a keep-alive message towards the TRAXIS.SRM

*) Note that in case of an OpenTV client (i.e. as identified in RTSP SETUP request by User-Agent = OpenTV VOD 1) the client shall denote the Content-Type header with the "Accept" phrase.

2.7.6 RTSP GET_PARAMETER Response

RTSP/1.0 200 OK - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
CSeq	No	Number
Session	No	Session identification @ TRAXIS.SRM
Content-Type	No	text/parameters
Content-Length	No	Length of RTSP request

2.7.7 RTSP ANNOUNCE message

TRAXIS.SRM shall only generate RTSP ANNOUNCE message in case TCP is used as the underlying network protocol. TRAXIS.SRM shall not send ANNOUNCE message in case UDP is used for the network protocol.

`ANNOUNCE * RTSP/1.0` – The * is used to denote the current presentation.

Header	Opt.	Usage
CSeq	No	Number
Session	No	Session identification @ TRAXIS.SRM
Notice	No	<Event-Code> <Event-Phrase> Event-date=<utc-time> Where • <Event-Code> → identifies the announcement • <Event-Phrase> → textual description of the announce code • <utc-time> → date/time of announcement in UTC[2]

Based on the event_code value STB shall

- 1000 – 1999 → Informational
- 2000 – 2999 → Informational which may result in a STB follow-up action
- 3000 – 3999 → Reserved (not used)
- 4000 – 4999 → Error related to either the streaming session or the asset
- 5000 – 5999 → Internal SRM error

Well-known announce-code can be defined but all announce_codes must be mapped accordingly to the above categories.

Announce code	Description
2401	Entitlement expired: The play-out has stopped.
2403	Session torn down: Heart beats missed.
2404	Session torn down: Asset has been forcefully deleted on server.
2406	Session torn down: Any other reason.
2407	Heart beats missed
5502	Internal Server error: the play out has stopped.

[2] Formatting of date/time according to chapter 1.6.

2.7.8 RTSP ANNOUNCE Response

`RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ TRAXIS.SRM

2.8 STB – Streaming server RTSP message syntax

Please refer to the appropriate documentation describing this interface. This interface might differ per on-demand client, Streaming server and/or deployment. At a minimum the RTSP PLAY and RTSP PAUSE message shall be supported. It is advisable also to provide support for the RTSP GET_PARAMETER for the retrieval of session related information (e.g. like position in stream) and the RTSP ANNOUNCE message as a mechanism to send announces generated by the Streaming server to the STB (e.g. end-of-stream reached) but this to be determined between STB and Streaming server.

2.8.1 RTSP PLAY Request

As per [RFC_2326]: `PLAY * RTSP/1.0` - the PLAY command uses the '*' character for URI to match the current presentation.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server
`Range`	Yes	The time-range to be played. If header is omitted the complete asset shall be played from start till finish, unless the stream is paused in which case the play-back continues from the point where it was paused.
`Scale`	Yes	Play-speed. A value of 1 indicates normal play-speed. If the Scale header is omitted normal play-speed will be applied.

Note that restrictions and/or extensions may apply based on support provided by on-demand client and/or streaming platform (e.g. Scale and Range format supported).

2.8.2 RTSP PLAY Response

As per [RFC_2326]: `RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server
`Range`	Yes	The actual range value as applied by the Streaming server
`Scale`	Yes	The actual scale value as applied by the Streaming server
`Date`	Yes	Timestamp of the response

Note that restrictions and/or extensions may apply based on support provided by on-demand client and/or streaming platform.

2.8.3 RTSP PAUSE Request

As per [RFC_2326]: `PAUSE * RTSP/1.0` - the PLAY command uses the '*' character for URI to match the current presentation.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server

Note that restrictions and/or extensions may apply based on support provided by on-demand client and/or streaming platform.

2.8.4 RTSP PAUSE Response

As per [RFC_2326]: `RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server
`Range`	Yes	The actual range value as applied by the Streaming server
`Scale`	Yes	The actual scale value as applied by the Streaming server
`Date`	Yes	Timestamp of the response

Note that restrictions and/or extensions may apply based on support provided by on-demand client and/or streaming platform.

2.8.5 RTSP GET_PARAMETER Request

As per [RFC_2326]: `GET_PARAMETER * RTSP/1.0` – the GET_PARAMETER command uses the '*' character for the URI, to match the current presentation.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server
`Accept/ Content-Type *)`	No	`text/parameters`
`Content-Length`	No	Length of RTSP request

Message body	Usage
…	Please refer to appropriate Streaming server specification for an overview of the supported GET_PARAMETER parameters (i.e. which parameter value can be retrieved like: Position in stream, Scale etc.).

*) Note that in case of an OpenTV client (i.e. as identified in RTSP SETUP request by User-Agent = `OpenTV VOD 1`) the client shall denote the Content-Type header with the "Accept" phrase.

2.8.6 RTSP GET_PARAMETER Response

As per [RFC_2326]: `RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server
`Content-Type`	No	`text/parameters`
`Content-Length`	No	Length of RTSP request

Message body	Opt.	Usage
…	No	Please refer to appropriate Streaming server specification for an overview of the supported GET_PARAMETER parameters (i.e. which parameter value can be retrieved like: Position in stream, Scale etc.)

2.8.7 RTSP ANNOUNCE message

The streaming server shall only generate RTSP ANNOUNCE message in case TCP is used as the underlying network protocol. The streaming server shall not send ANNOUNCE message in case UDP is used for the network protocol.

As per [RFC_2326]: `ANNOUNCE * RTSP/1.0` – The * is used to denote the current presentation.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server
…	No	Please refer to appropriate Streaming server specification for an overview of the supported ANNOUNCE headers

2.8.8 RTSP ANNOUNCE Response

As per [RFC_2326]: `RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	No	Session identification @ Streaming server

3 TRAXIS.SRM → Streaming server

This section defines the interface between the SRM and the Streaming server for the setup and teardown of streaming sessions at the Streaming server. The message protocol (and mechanisms/semantics) is based on the RTSP protocol.

The following overall assumptions apply:

- The on-demand client addresses the SRM at the initiation of the streaming session (i.e. RTSP SETUP); the SRM is the sole entry point into the on-demand system for the client it acts as the single “proxy” for all streaming servers located in the network. The client also addresses the SRM to terminate the streaming session (i.e. RTSP TEARDOWN). The way the STB is provisioned the location/address of the SRM is out-of-scope.
- Based on the client’s location in the network (i.e. ServingGroup) the SRM manages the allocation of the Streaming and Edge resources. Note that the SRM is logical entity responsible for both the Session and the Resource management; the interaction between the SM and the RM is out-of-scope.
- The client addresses the streaming control messages directly towards the Streaming server (i.e. RTSP PLAY, RTSP PAUSE, …)
- Whether or not a streaming session is setup or torn down at the Streaming server is fully managed by TRAXIS.SRM. The Streaming server must never setup or teardown a session if it is not instructed to do by the TRAXIS.SRM. The Streaming server slaves to the TRAXIS.SRM

3.1 TCP-connections

TRAXIS.SRM shall create a single TCP-connection with each of the Streaming servers whose streaming resources are managed by the TRAXIS.SRM. All individual session setup and teardown requests shall be addressed to the Streaming server via this single TCP-connection.

TRAXIS.SRM (i.e. client) is responsible for the TCP-connection. In case of a connection failure TRAXIS.SRM shall re-establish the TCP-connection between TRAXIS.SRM and the Streaming server. The Streaming server shall take no action in case the TCP-connection is broken. Optionally there shall be keep-alive messages between the TRAXIS.SRM and the Streaming server at application level (i.e. RTSP GET_PARAMETER) to avoid the explicit closure of the connection by the Streaming server due to an inactivity time-out.

Although the TRAXIS.SRM operates as a single logical entity, it may be deployed as multiple physical systems (e.g. in case of a redundant deployment of TRAXIS.SRM for load sharing reasons). Each of the individual physical systems shall maintain a permanent TCP- connection with each of the Streaming servers managed by the TRAXIS.SRM implying that an individual Streaming server should be able to accept and maintain multiple connections with each the individual physical systems. The Streaming server shall not assume that the same TCP-connection will be used to address a particular session.

In order for the Streaming server to differentiate between the physical TRAXIS.SRM systems, the first RTSP GET_PARAMETER message issued by a physical system after the TCP-connection is established shall contain a body with an identification of the physical system (i.e. the machine name of the system).

3.2 UDP

UDP as delivery protocol between TRAXIS.SRM and the streaming server is not supported. In case UDP is used as delivery protocol between the STB and TRAXIS.SRM, the streaming server platform TCP is used for the messaging between TRAXIS.SRM and the streaming server.

3.3 TRAXIS.SRM – Streaming server synchronization

In order to avoid the built-up of orphan streams at the Streaming server and/or orphan resource allocations at the TRAXIS.SRM, TRAXIS.SRM will issue a re-synchronization session at a regular interval (e.g. once a day at 04:00 hours) to synchronize the number of active streams between the Streaming server and TRAXIS.SRM.

1. TRAXIS.SRM requests the number of (active) streams at the Streaming server
2. If the number of (active) streams coincides between the Streaming server and the TRAXIS.SRM
    a. No actions required from TRAXIS.SRM
3. If number of streams at Streaming server <> number of streams at TRAXIS.SRM
    a. TRAXIS.SRM requests a list of all active streams at Streaming server
    b. TRAXIS.SRM compares the list with its own set of active streams
    c. If number of streams at Streaming server > number of streams at TRAXIS.SRM
        i. Only TRAXIS.SRM controls the setup of the streams at the Streaming server, therefore all additional streams active at the Streaming server must already been torn down by the on-demand client at the SRM, but not yet at the Streaming server by TRAXIS.SRM. ➔ For all streams active at the Streaming server but not at the SRM, the SRM issues a tear down requests, upon which the Streaming server shall tear down the streaming session and, if required close the TCP-connection with the STB.
    d. If number of streams at Streaming server < number of streams at TRAXIS.SRM
        i. Somehow streams are torn down at the Streaming server without the TRAXIS.SRM issuing a tear down for these sessions. ➔ All allocations for streams not active at the Streaming server are released by TRAXIS.SRM. These allocations are considered to be false allocations.

The re-synchronization mechanism is also executed in case of a connection failure between the Streaming server and the TRAXIS.SRM to inform the Streaming server of any stream tear down occurred during the time of the connection failure.

3.4 TCP Connection timeout and recovery

There shall be no monitoring of the individual sessions between the TRAXIS.SRM and the Streaming server. However it is the responsibility of the TRAXIS.SRM to maintain the TCP-connection and in case the connection is lost to re-establish the TCP-connection.

The Streaming server may decide to close the connection after connectivity time-out – i.e. if the Streaming server did not receive a message from the SRM after a (configurable) time-out. Any message send by the SRM shall reset the time-out. In case of overall system inactivity (i.e. the SRM did not receive a session setup or teardown request from any on-demand client), the SRM may send a connection keep-alive message to prevent the Streaming server from closing the connection (i.e. a RTSP GET_PARAMETER message).

3.4.1 TRAXIS.SRM recovery

0. TRAXIS.SRM failure → TCP-connection between TRAXIS.SRM and Streaming server is lost
1. Streaming server detects loss of TCP-connection
    a. Streaming server maintains the currently active streaming sessions
2. TRAXIS.SRM is recovered → TCP-connection between TRAXIS.SRM and Streaming server is established
3. Streaming server accepts the newly established TCP-connection
4. TRAXIS.SRM resets all on-demand client session timeouts

5. TRAXIS.SRM will synchronize the sessions at the Streaming server with the session as known at TRAXIS.SRM
    a. All active sessions and the corresponding resource allocations are persisted at the TRAXIS.SRM, which implies that under normal circumstances there is no need for the synchronization TRAXIS.SRM and Streaming server.

3.4.2 Streaming server recovery or TCP-connection recovery

TRAXIS.SRM has no means to distinguish between a Streaming server failure and a TCP-connection failure, in both cases the TCP-connection is lost.

0. TCP-connection between TRAXIS.SRM and Streaming server is lost
1. TRAXIS.SRM tries to re-establish at regular intervals the TCP-connection (until Streaming server or TCP-connection is recovered)
2. Streaming server accepts the newly established TCP-connection
3. TRAXIS.SRM will synchronize the sessions at the Streaming server with the session as known at TRAXIS.SRM

3.5 TRAXIS.SRM – Streaming server RTSP message syntax

The RTSP messages follow in syntax, semantics and operations according [RFC_2326].

3.5.1 Support for additional features (protocol versioning)

In order for the SRM to identify its capabilities and the expected Streaming server capabilities the RTSP messages as exchanged between the SRM and Streaming server may include a "Require" header. Via the "Require" header the SRM requests support for additional features beyond the minimal set of required features.

Upon a feature request not supported by the Streaming server the Streaming server should respond with a "`551 Option not supported`" where the unsupported feature(s) is (are) listed with the aid of the "Unsupported" header in the response returned by the Streaming server (see [RFC_2326] for a detailed description on the usage of the "Require" and "Unsupported" headers).

The table below presents an overview of the additional features beyond the basic set of features supported by the SeaChange back-office.

Require tag	Description
…	…

3.5.2 RTSP SETUP Request

```
SETUP rtsp://<host>:<port>/<asset> RTSP/1.0
```

If one or more playlist items are included in the body of the message the `<asset>` in the RTSP URI must be ignored. The playlist it`em`(s) overrule the `<asset>` definition in the URI.

Per playlist item is identified which trick-modes are disallowed (i.e. PAUSE, FF and REW), the Streaming server is expected to honor these trick-mode restrictions (per playlist item) upon a "trick-mode request" from the STB during the play-out of the asset play-list. The following behavior is expected:

- If PAUSE, FF or REW is requested during the play-out of a playlist item where the requested trick-mode is not allowed the Streaming server should respond with a `405: Not allowed` towards the STB
- If FF is activated during the play-out and the stream position reaches an item where FF is disallowed, the Streaming server should "fall back" to normal PLAY mode (i.e. PLAY with scale == 1.0).
- If REW is activated during the play-out and the stream position reaches an item where REW is disallowed, the streaming server should skip the playlist item for which REW is disallowed and continue with first previous item in the playlist for which REW is allowed.
- The Streaming server should not allow the client to jump ("forward jump") over a play-list item for which FF is disallowed. If FF is activated and the playlist reaches a playlist item for which FF is forbidden, the Streaming server should fall back to normal PLAY mode (i.e. PLAY with scale == 1.0) at the beginning of the playlist item for which FF is disallowed.
  However, to allow previous sessions to be resumed, the Streaming server should allow the first client PLAY request for a newly created streaming session to jump ("forward jump") over or into a playlist item for which FF is disallowed, provided the request is for normal PLAY mode (i.e. PLAY with scale == 1.0). If this first client PLAY request for a newly created streaming session contains a different speed than 1.0, the Streaming server should fall back to normal PLAY mode (i.e. PLAY with scale == 1.0) but honor the jump request to that position.
- The Streaming server should allow a "backward jump" over a playlist item for which REW is disallowed, but is should allow to jump into a playlist for which REW is forbidden. If the to-jump position is located within an item for which REW is disallowed the Streaming server should jump to first previous playlist for which REW is allowed.

Header	Opt.	Usage
`CSeq`	No	Number
`Require`	Yes	Additionnelly requested feature support
`Transport`	No	`MP2T/DVBC/UDP;unicast;client=`<client>`;source=`<source>`;` `destination=`<destination>;`client_port=`<client_port> Where • <client> → On-demand client identification (for logging purposes only) • <source> → Stream source IP address (optionally including a subnet mask) at the streaming server • <destination> → Stream destination IP address (i.e. in DVB-C the EdgeQAM and for IPTV the IP address of the STB) • <client_port> → Stream destination port number If the Stream source is presented including a subnet mask, the Streaming server can use any IP-address within the subnet range as a source address for the stream. For example 192.168.1.0/28 implies that all IP-addresses in the range 192.168.1.1 till 192.168.1.14 can be used.
`Content-Type`	Yes	`text/parameters`
`Content-Length`	Yes	Length of RTSP request body

Message body	Opt.

playlist_item:<asset>;ordinal=<ordinal>[;in=<NPT_in>][;out=<NPT_out>][;trickmode=<trickmode>] Where • <ordinal> → position of the asset in the playlist • <NPT_in> → The offset in NPT to start playing the asset from (NPT conform [RFC_2326]). • <NPT_out> → The last frame in NPT of the asset that is playable by the VOD Server. • <trickmode> → A decimal representation of the bitmask value specifying the trick play mask to be applied to this asset If this value is not supplied or the value equals 0, all trick play modes are supported. Individual flags in the bit mask: ○ 0x01 → PAUSE disallowed ○ 0x02 → FF disallowed ○ 0x04 → REW disallowed ○ 0x08 → PLAY disallowed (not yet implemented) ○ 0x10 → SKIP disallowed Multiple playlist items can be presented to indicate the individual items of the playlist each with its characteristics (e.g. allowed trickmodes)	Yes

3.5.3 RTSP SETUP Response

RTSP/1.0 200 OK - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure. Note that if a playlist item/asset requested is unavailable/unknown at the Streaming server, the Streaming server should not setup a session for the requested playlist and return a “Content not found error (i.e. 404).

Header	Opt.	Usage
CSeq	No	Number
Session	No	Session identification @ Streaming server
Transport	Yes	MP2T/DVBC/UDP;unicast;client=<client>;destination=<destination>;client_port=<client_port>[;source=<source>;server_port=<server_port>] This shall match the requested transport parameters of the request, optionally extended with the source IP-address and port number of the stream (i.e. source at the Streaming Server)[3].
Location	Yes	Stream control RTSP URI In the absence of the Location header TRAXIS.SRM will substitute the default value in the response towards the on-demand client.
Content-Type	Yes	text/parameters
Content-Length	Yes	Length of RTSP request

[3] In case of an IPTV deployment, the stream source information will be forwarded in the RTSP SETUP response towards the IPTV client devices, to allow the IPTV client to do a “NAT hole punching” if required.

Message body	Opt.
playlist_item:<asset>;ordinal=<ordinal>[;in=<NPT_in>][;out=<NPT_out>] [;trickmode=<trickmode>];nptlength=<nptlength> Multiple playlist items can be presented to indicate the individual items of the playlist including per play-list item the length of the item. In case a requested playlist asset is not available/unknown at the Streaming server, the Streamer server must return an empty value (i.e. nptlength=).	Yes

Note that the playlist items in the body of the response only required is if a playlist is presented in the request.

3.5.3.1 Redirection

In case the asset storage of the local Streaming server does not contain a copy of the asset, but the asset is available within the Streaming server platform, the local Streaming server may respond with a redirection response: `RTSP/1.0 303 See Other`.

Header	Opt.	Usage
CSeq	No	Number
Location	No	rtsp://<host>:<port>/<asset>[, rtsp://<host>:<port>/<asset>] Comma separated list of URI-s indicating the location of the requested asset.

3.5.4 RTSP TEARDOWN Request

`TEARDOWN * RTSP/1.0` - the TEARDOWN command uses the '*' character for URI to match the current presentation.

Header	Opt.	Usage
CSeq	No	Number
Require	Yes	Additionnelly requested feature support
Session	No	Session identification @ Streaming server
Reason	Yes	Reason for the teardown – can be used for logging purposes

3.5.5 RTSP TEARDOWN Response

`RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
CSeq	No	Number
Date	Yes	Timestamp of the response

Message body	Opt.
current_npt=<current_npt> This is the current npt of the stream when the session was torn down, measured as across the	Yes

aggregated nptlength of the individual playlist items.	
in=<NPT_in>;out=<NPT_out>;scale=<scale>;start_time=<start_time>; end_time=<end_time> Following the current_npt there may be one or more stream control action lines (each with a proper line termination as defined in [RFC_2326]). Together they indicate the session history during playback in the session that was torn down. Each line denotes a particular stream control action, being a period when a certain scale factor was applied or in case of 'skip' the modified stream position. This implies that at least one line will appear in the history (for the 'play' without any other trickmode). Parameters: • <NPT_in>, <NPT_out> → The offsets in NPT relative to the aggregated playlist that indicate where the scale factor was applied (NPT conform [RFC_2326]). • <scale> → Float denoting the stream scale factor used in the period indicated by start_time and end_time. • <start_time>, <end_time> → Start and end time in UTC of the period during which the scale factor was applied. [4] This field is optional. It is not filled if the video server does not support logging of trick history. Note that if not filled, corresponding session reporting towards e.g. an advertising system cannot take place. Note that the current_npt shall always precede the trick history line items and should also be properly line terminated as defined in [RFC_2326].	Yes

3.5.6 RTSP GET_PARAMETER Request

GET_PARAMETER * RTSP/1.0 – the GET_PARAMETER command uses the '*' character for the URI, to match the current presentation.

Header	Opt.	Usage
CSeq	No	Number
SessionRequire	Yes	Session identification @ Streaming server if applicable. Session is not applicable in case of a session_count or session_list request.Additionally requested feature support
Session	Yes	Session identification @ Streaming server if applicable. Session is not applicable in case of a session_count or session_list. request or when the GET_PARAMETER message is used as a keep-alive message (i.e. GET_PARAMETER request with an empty body)
Content-Type	No	text/parameters
Content-Length	No	Length of RTSP request body

[4] Formatting of date/time according to chapter 1.6.

Message body	Opt.	Usage
`<<empty>>`	N/A	Used as a keep-alive message towards the streaming server. The first keep-alive message send by the SRM after the TCP-connection is established shall contain a body containing the text: `SRM_ID: <SRM machine name>`
`session_count`	No*	Request for number of active sessions
`session_list`	No*	Request for list of active
`stream_state`	Yes	Request for the current state of the stream
`Position`	Yes	Request for current stream position
`Scale`	Yes	Request for current stream scale

*) Either session_count **or** session_list shall be provided. TRAXIS will request a session_count first and if there is a mismatch with the records in the SRM it will issue a session_list request.

3.5.7 RTSP GET_PARAMETER Response

`RTSP/1.0 200 OK` - or appropriate error status (i.e. status-code and reason phrase describing the status-code) in case of failure.

Header	Opt.	Usage
`CSeq`	No	Number
`Session`	Yes	Session identification @ Streaming server if applicable. Session is not applicable in case of a `session_count` or , `session_list request` or when the `GET_PARAMETER` message is used as a keep-alive message (i.e. `GET_PARAMETER` request with an empty body)
`Content-Type`	No	`text/parameters`
`Content-Length`	No	Length of RTSP request body

Message body	Opt.	Usage
`session_count`	No*	Integer depicting the number of active sessions. `session_count: <session_count>`

`session_list`	No*	List of active session identifications according to syntax:
		`Session_list: <identification @ Streaming server>`
		`<identification @ Streaming server>*`
		Multiple entries are separated with a white-space.

`stream_state`	Yes	Possible values are: • init • ready • playing • paused • stopped • stalled

		stream_state: <“init”|“ready”|“playing”|”paused”|”stopped”|”stalled”>
Position	Yes	Current stream position in NPT Position: <position>
Scale	Yes	Float denoting the current stream scale Scale: <scale>

*) Only mandatory when requested in request message.

3.5.8 RTSP ANNOUNCE message

The Streaming server is expected (but not required) to send a RTSP ANNOUNCE message to inform the TRAXIS.SRM about the status of an individual session.

Any ANNOUNCE messages send by the Streaming server to the TRAXIS.SRM should be delivered at all connected (physical) systems via the multiple TCP-connection established between the Streaming server and the TRAXIS.SRM. In case the Streaming server is unable to deliver the ANNOUNCE message the Streaming server is not expected to retry the delivery of the ANNOUNCE message.

`ANNOUNCE * RTSP/1.0` – The * is used to denote the current presentation.

Header	Opt.	Usage
CSeq	No	Number
Session	No	Session identification @ Streaming server
Notice	No	<Event-Code> <Event-Phrase> Event-date=<utc-time> Where • <Event-Code> → identifies the announcement • <Event-Phrase> → textual description of the announce code • <utc-time> → date/time of announcement in UTC[5]

Based on the event_code value TRAXIS.SRM shall

- 1000 – 1999 → Informational
- 2000 – 2999 → Informational which may result in a TRAXIS.SRM follow-up action
- 3000 – 3999 → Reserved (not used)
- 4000 – 4999 → Error related to either the streaming session or the asset
- 5000 – 5999 → Internal Streaming server error

Well-known announce-code must be defined but all announce_codes must be mapped accordingly to the above categories.

[5] Formatting of date/time according to chapter 1.6.

Announce code	Description
4400	Error Reading Content Data: the play out has stopped.
5200	Server Resources Unavailable: The play out has stopped.
5402	Client Session Terminated: teardown has been initiated by the server, the session has been torn down.
5403	Server Shutting Down: the play out has stopped.
5502	Internal Server error: the play out has stopped.

3.5.9 RTSP ANNOUNCE Response

`RTSP/1.0 200 OK` - or appropriate error status in case of failure

Header	Opt.	Usage
CSeq	No	Number

4 Stream Control proxy

In order to provide support for RTSP clients, who do not provide support for the RTSP protocol (as described in this specification), where the session control messages and the stream control message are delivered at respectively the SRM and the Streaming server. TRAXIS.SRM is also able to run in a “Stream Control proxy” mode.

In “Stream Control proxy” mode TRAXIS.SRM is the end-point for all RTSP messages. TRAXIS.SRM acts as a proxy for the RTSP PLAY, RTSP PAUSE and RTSP GET_PARAMETER messages normally send to the Streaming server.

4.1 Session setup / teardown

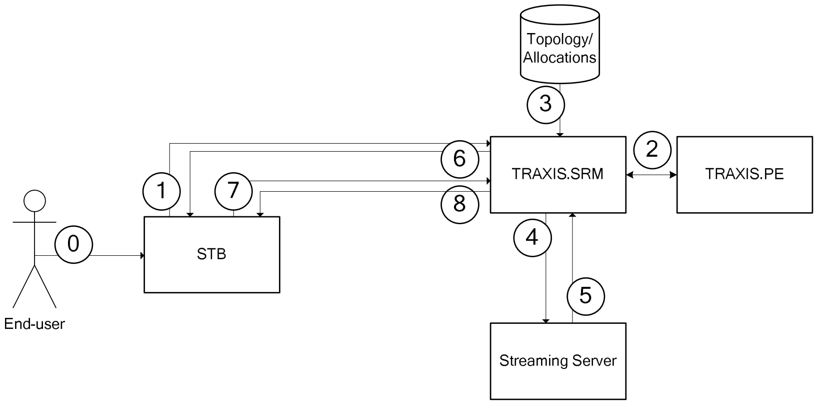

FIGURE 3: STREAMING SESSION SETUP

The figure above depicts the message exchange during a streaming session setup.

- The end-user initiates the setup of the streaming session for a particular asset via the on-demand-client (0)
- The on-demand client addresses the session manager (i.e. TRAXIS.SRM) with the request to setup a streaming session (1) → RTSP SETUP @ SRM
- TRAXIS.SRM optionally authorizes the request to setup a streaming session for the requested asset with TRAXIS.PE (2). TRAXIS.PE is the Purchase & Entitlement Manager and maintains the on-demand entitlements per end-user [TRAXIS_PE].
- Based on the serving area of the requesting STB, TRAXIS.SRM determines for which Streaming server and Edge-QAM to allocate a stream (3).
  - The Edge-QAM allocation is only applicable in a DVB-C deployment.
- TRAXIS.SRM instructs the allocated Streaming server to setup a stream
  - In a DVB-C deployment → to the allocated Edge-QAM (4)
  - In an IPTV deployment → to the requesting STB (4)

  And the Streaming server returns the session identification (5) → RTSP SETUP @ Streaming server.
- Upon a successful session setup on the Streaming server, TRAXIS.SRM returns towards the STB :
  - the session identification of the session setup by the STB
- The on-demand client issues a start streaming command at the Streaming server (7) and TRAXIS.SRM responds appropriately (8) → RTSP PLAY @ SRM
- Once the streaming has started the on-demand client is expected to send at regular intervals a keep-alive message to TRAXIS.SRM. In the absence of the keep-alive message TRAXIS.SRM may decide to tear down the streaming session → RTSP GET_PARAMETER @ SRM

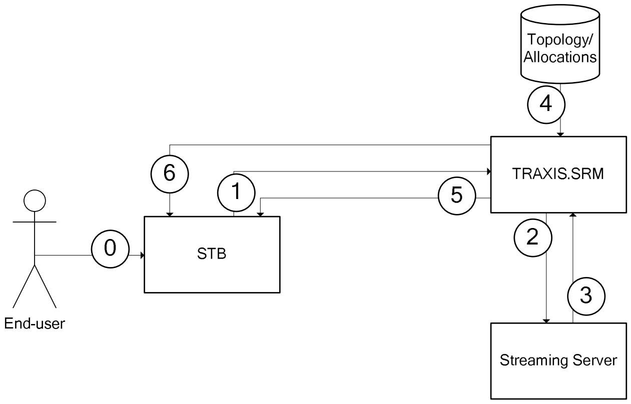

FIGURE 4: STREAMING SESSION TEARDOWN

The streaming session teardown message exchange includes:

- The end-user initiates the teardown of the session via the STB's on-demand client (0)
- The on-demand client addresses TRAXIS.SRM to tear down the session at the TRAXIS.SRM (1)
- TRAXIS.SRM instructs the Streaming server to tear down the corresponding session at the Streaming server (2). After the Streaming server torn down the session it shall respond with the confirmation of the session teardown (3)
- TRAXIS.SRM releases the Streaming server resources and in a DVB-C deployment also the Edge-QAM resources allocated to the session (4) and inform the on-demand client that the session has been torn-down (5).

Note that, upon a keep-alive time-out, a streaming session teardown may also be initiated by TRAXIS.SRM

- TRAXIS.SRM informs the on-demand client that the streaming session will be torn down (6). The on-demand client may respond to the announcement but TRAXIS.SRM does not expect back to receive a response.
- TRAXIS.SRM instructs the streaming server to tear down the session (2), (3) and releases the allocated session resources (4).

4.2 TCP-connection / UDP

All RTSP message are directed to TRAXIS.SRM, therefore

- TCP ➔ the STB only maintains a single TCP-connection with the TRAXIS.SRM.
- UDP ➔ all RTSP request are send to TRAXIS.SRM

4.3 STB – TRAXIS.SRM RTSP message syntax

4.3.1 RTSP SETUP Request / Response

The RTSP SETUP Requestrequest messages is equal to [2.7.1] and TRAXIS.SRM handles the RTSP SETUP message request from a STB in a regular way (i.e. TRAXIS.SRM instruct the Video Server to setup a stream to be delivered at the end-point as depicted in the request ...). However the response message is equal to [2.7.2] except that the response message does not contain the `SessionControl` header and the `Location` header.

4.3.2 RTSP PLAY Request / Response

See [2.8.1]The RTSP PLAY message request presented by the STB at TRAXIS.SRM is forwarded by TRAXIS.SRM as is to the streaming server and the response from the streaming server is forwarded by TRAXIS.SRM to the requesting STB ([2.8.1], [2.8.2]).

4.3.3 RTSP PAUSE Request / Response

See [2.8.3]The RTSP PAUSE message request presented by the STB at TRAXIS.SRM is forwarded by TRAXIS.SRM as is to the streaming server and the response from the streaming server is forwarded by TRAXIS.SRM to the requesting STB ([2.8.3], [2.8.4]).

4.3.4 RTSP TEARDOW Request / Response

See The RTSP TEARDOWN request messages is equal to [[2.7.3] and TRAXIS.SRM handles the RTSP TEARDOWN message request from a STB in a regular way (i.e. TRAXIS.SRM instruct the Video Server to teardown the stream). The response message is equal to [2.7.4].

4.3.5 RTSP GET_PARAMETER Request / Response

All RTSP GET_PARAMETER messages delivered by the STB at TRAXIS.SRM are forwarded towards the streaming server except for the GET_PARAMETER messages which are used as a keep-alive message (i.e. GET_PARAMETER message with an empty body) See [2.7.5] and [2.7.6] for an overview of the GET_PARAMETER messages and their usage.

4.3.6 RTSP ANNOUNCE Request / Response

See RTSP ANNOUNCE messages send by the streaming server to inform the STB about a particular condition that has arisen are forwarded by TRAXIS.SRM to the appropriate STB ([2.7.7] and], [2.7.8]).

5 Notifying TRAXIS.SRM of Sessions Events

In order to provide support of session logging and reporting in case TRAXIS.SRM is not used or it concerns progressive download, a SessionNotification HTTP call is introduced.

5.1 Session Notification for Progressive Download deployments

For the progressive download deployment explanation, see 5g. The SessionNotification call described below is the same for progressive download deployments.

5.2 Session Notification for non TRAXIS.SRM deployments

5.2.1 Session Setup

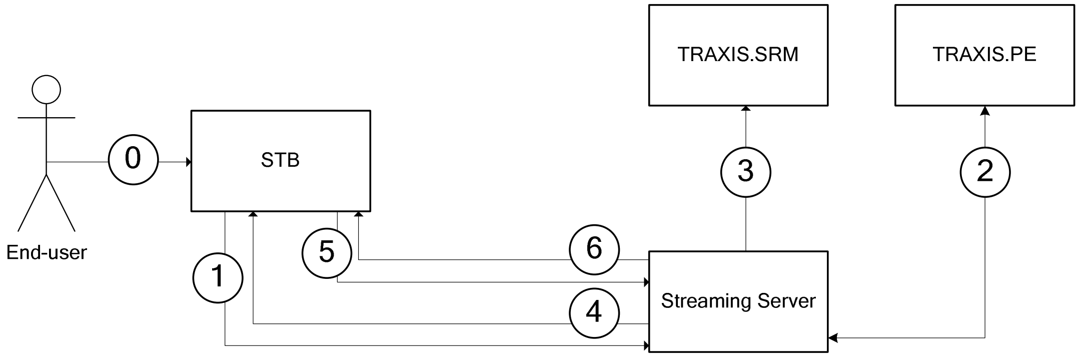

FIGURE 5: SESSION NOTIFICATION AT SETUP

The figure above depicts the message exchange during a streaming session setup.

- The end-user initiates the setup of the streaming session for a particular asset via the on-demand-client (0)
- The on-demand client addresses the session manager/streaming server with the request to setup a streaming session (1)
- The session manager/streaming server optionally checks the authorization at TRAXIS.PE (2)
- The session manager/streaming server sends a session notification "start" to TRAXIS.SRM (3)
- The session manager/streaming server returns OK (4)
- The on-demand client issues a start streaming command at the streaming server (5) and the streaming server responds appropriately (6)

5.2.2 Session Teardown

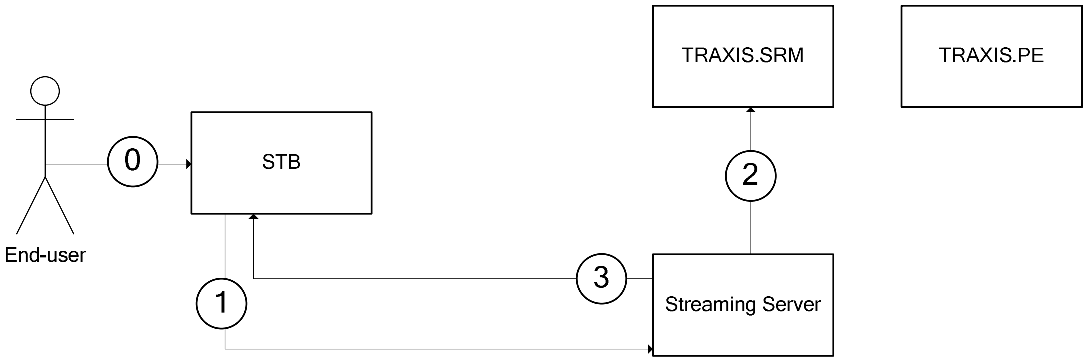

FIGURE 6: SESSION NOTIFICATION AT TEARDOWN

The streaming session teardown message exchange includes:

- The end-user initiates the teardown of the session via the STB's on-demand client (0)
- The on-demand client addresses the session manager/streaming server to tear down the session (1)
- The session manager/streaming server sends a session notification "end" to TRAXIS.SRM (2)
- After the Streaming server has torn down the session it shall respond with the confirmation of the session teardown (3)

5.3 SessionNotification.traxis

The following syntax is used to notify TRAXIS.SRM on session changes.

```
GET
 /SessionNotification.traxis?StbId=<stbId>&AssetId=<assetId>&SessionId=<sessionId>&Sess
 ionType=<sessionType>&EventCode=<eventCode>[&EventReason=<eventReason>][&EventDate=<ev
 entDate>][&Bookmark=<bookmark>] HTTP/1.1
```

Parameters

- string *StbId*: (Required) The identification of the client device.
- string *AssetId*: (Required) The identification of the asset to download.
- string *SessionId*: (Required) The identification of the session. The session must be unique in time and per set-top box.
- string *SessionType*: (Required) The delivery method of the session. Possible values are:
    - "PDL": Delivery via PDL (Progressive Download) to the Stb.
    - "QAM": Delivery via QAM streaming to the Stb.
    - "IPTV": Delivery via IP streaming to the Stb.
- string *EventCode*: (Required) The type of notification of the session. Possible values are:
    - "start": The session has started.
    - "end": The session has ended.
- string *EventReason*: (Optional) Describes the reason for the notification. The format is free to pick, maximum of 50 characters.

- string *EventDate*: (Optional) The date and time the event occurred. (Format in UTC time according to ISO8601). This parameter can be used for offline processing of session events.
- string *Bookmark:* (Optional) The current NPT value of the play back position. (NPT conform [RFC2326]). It is only allowed to send the bookmark value, when using the EventCode “end”.

Remarks

The sessionId must be unique in time and per set-top box. Suggest to use a format like: <stbId>-yyyyMMddHHmmss. The same sessionId used for the “start” event, must be used at the “end” event of that session.

Traxis will not verify the existence of the parameters in its database. As such Traxis will not fail if for example an invalid StbId or AssetId is given. The reason for this are two-fold, Traxis might not have the information or in case of the offline method, the information could already have been removed from Traxis.

In case the set-top box is unable to send the SessionNotification “end”, either the end will be forced by a specified timeout or when the set-top box sends another SessionNotification “start”.

Headers

Header	Opt.	Usage
`Host`	No	Required parameter for HTTP/1.1: The Host request-header field specifies the Internet host and port number of the resource being requested, as obtained from the original URI given by the user or referring resource. A "host" without any trailing port information implies the default port for the service requested (e.g., "80" for an HTTP URL).

HTTP Request

The following is a sample HTTP GET request. The placeholders [...] and {...} need to be replaced with the actual values.

```
GET /SessionNotification.traxis?StbId={string}&AssetId={string}&SessionId={string}
&SessionType={string}&EventCode={string}&EventReason={string}&EventDate={string}
&Bookmark={string} HTTP/1.1
Host: Traxis.srm
```

HTTP Response

The following is a sample response. The placeholders [...] and {...} need to be replaced with the actual values.

```
HTTP/1.1 200 OK
```

Error codes

The following error codes could be returned by Traxis:

HTTP Error code	Error Message	Reason
400 Bad Request	EInvalidStbId EInvalidAssetId EInvalidSessionId EInvalidSessionType EInvalidEventCode EInvalidBookmark	The StbId is not known The AssetId is not known The SessionId is not known (“end” only) SessionType is invalid EventCode is invalid Bookmark was invalid

	ENotEntitled EMissingParameter	The Stb was not authorized for Asset Required parameter missing
500 Internal Server Error		Something unexpected happened.

The standard errors that are reported back from the system have the following HTTP response body: *ErrorCode=<error>*.

Examples:

An expected parameter was missing:

```
HTTP/1.1 400 Bad Request
Content-Length: 27
Content-Type: text/plain; charset=utf-8
Server: 3.0.0.1279 Microsoft-HTTPAPI/2.0
Date: Tue, 20 Apr 2010 13:32:11 GMT

ErrorCode=EMissingParameter
```

The Stb was not authorized:

```
HTTP/1.1 400 Bad Request
Content-Length: 22
Content-Type: text/plain; charset=utf-8
Server: 3.0.0.1279 Microsoft-HTTPAPI/2.0
Date: Tue, 20 Apr 2010 13:33:09 GMT

ErrorCode=ENotEntitled
```

6 On-demand network topology and allocation

6.1 DVB-C deployment topology

TRAXIS.SRM needs to be aware of the deployed on-demand network topology for the determination of the Streaming server and, in a DVB-C deployment, the Edge-QAM for the delivery of the STB requested stream. TRAXIS.SRM maintains (a copy of) the configuration of network topology as it is defined network operator.

The main components of the on-demand network topology include:

- The Streaming servers
    - Management port (IP-address and port number)
    - Streaming control URI
    - The number and identification of the streaming ports
        - IP-address (streaming source address)
        - The streaming capacity of the Streaming server (max. output bit-rate or max. number of streams)
        - QAM input IP-addresses (streaming destination address) and weighting factor.
- The Service groups
    - Service group identification – this coincides with the Service group identification presented by the STB on-demand application at the session setup
    - QAM-channels serving the Service group
- DVB-C only, the Edge-QAMs, per edge-QAM chassis
    - Management port (IP-address and port number)
    - The number and identification of (IP-)input ports
        - IP-address, optionally including a subnet mask.
        - Maximum bit-rate and maximum number of streams (either max. bit-rate or max. number of streams defines input capacity)
    - The number and identification of the installed QAM-modules
    - Per QAM-module the number and identification of the QAM-outputs (all apply to all channels of the QAM-output)
        - Maximum bit-rate per channel
        - Modulation (QAM-265, QAM-64, etc.)
        - Symbol-rate
        - Inner FEC and outer FEC (optional)
    - Per QAM-output the number and identification of channels
        - Transport-stream ID
        - Center frequency
        - Maximum number of streams (either max. bit-rate or max. number of streams limits the number of streams per channel)
    - The IP-input → Frequency/program number mapping algorithm
    - Program number offset

6.2 IPTV deployment topology

<<TO DO>>

6.3 Topology configuration upload

The on-demand network topology is presented towards TRAXIS.SRM as an XML-file [TOP_XSD]. Upon upload of the topology XML-file TRAXIS.SRM will determine the difference between the topology it currently knows and the newly requested topology and update the topology accordingly.

For each of use (and recommended) the on-demand network topology is maintained by the network operator as an Excel sheet. Full details on how to use the Excel sheet are provided by [TOP_RM]. Tooling to validate and convert the Excel to the topology XML-file is available.