Open IPTV Forum Release 1 Specification

Volume 4 – Protocols

[V1.2] – [2012-08-28]

Reformatted 2012-09-21

Open IPTV Forum

Postal address

Open IPTV Forum support office
650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 43 83
Fax: +33 4 92 38 52 90

Internet

http://www.oipf.tv

Disclaimer

The Open IPTV Forum accepts no liability whatsoever for any use of this document.

This specification provides multiple options for some features. The Open IPTV Forum Profiles specification complements the Release 1 specifications by defining the Open IPTV Forum implementation and deployment profiles. Any implementation based on Open IPTV Forum specifications that does not follow the Profiles specification cannot claim Open IPTV Forum compliance.

Copyright Notification

Contents

Figures

Tables

Foreword

This Technical Specification (TS) has been produced by the Open IPTV Forum.

This specification provides multiple options for some features. The Open IPTV Forum Profiles specification complements the Release 1 specifications by defining the Open IPTV Forum implementation and deployment profiles. Any implementation based on Open IPTV Forum specifications that does not follow the Profiles specification cannot claim Open IPTV Forum compliance.

This document is Volume 4 in the 7 Volume set of specifications that define the Open IPTV Forum Release 1 Solution. Other Volumes in the set are:

- Volume 1 - Overview
- Volume 2 - Media Formats
- Volume 3 - Content Metadata
- Volume 5 - Declarative Application Environment
- Volume 6 - Procedural Application Environment
- Volume 7 - Authentication, Content Protection and Service Protection

Introduction

This document specifies the protocols over the following reference point interfaces defined in the Open IPTV Forum Release 1 Architecture specification [ARCH]

- The UNI interfaces, between the network or service provider domains and the consumer domain
- The HNI interfaces, between the functional entities in the consumer network domain
- The NPI interfaces, between the functional entities in the network and service provider domains
- Interfaces to external systems, which include
    - DLNA networks in the consumer domain

The requirements for these interfaces are derived from the following sources:-

- Open IPTV Forum Service and Platform Requirement for Release 1 [REQS]
- Open IPTV Forum Functional Architecture for Release 1 [ARCH]
- Other Open IPTV Forum specifications [DAE], [CSP], [META] and [MEDIA]

1 References

1.1 Normative References

[TS124503]	ETSI, TS 124 503, “Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); TISPAN; IP Multimedia Call Control Protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP) Stage 3” [3GPP TS 24.229 (Release 7), modified] (3GPP TS 24.503 Release 8)
[UMTS-SH]	ETSI, TS 129 329, “Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Sh interface based on the Diameter protocol; Protocol details” (3GPP TS 29.329 version 7.4.0 Release 7)
[CHNG]	ETSI, ES 282 010, “Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Charging”
[DIAM]	ETSI, TS 183 033, “Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN);IP Multimedia; Diameter based protocol for the interfaces between the Call Session Control Function and the User Profile Server Function/Subscription Locator Function; Signalling flows and protocol details” [3GPP TS 29.228 V6.8.0 and 3GPP TS 29.229 V6.6.0, modified]
[AFSPDF]	ETSI, TS 183 017, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN);Resource and Admission Control: DIAMETER protocol for session based policy set-up information exchange between the Application Function (AF) and the Service Policy Decision Function (SPDF);Protocol specification”
[RACS-RE]	ETSI, TS 183 060. “Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Resource and Admission Control Subsystem (RACS); Re interface based on the DIAMETER protocol”
[NASS-E4]	ETSI, ES 283 034, “Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN);Network Attachment Sub-System (NASS);e4 interface based on the DIAMETER protocol”
[RFC2119]	IETF, RFC 2119, “Keywords for use in RFCs to Indicate Requirement Levels”
[HTTP]	IETF, RFC 2616, “Hypertext Transfer Protocol -- HTTP/1.1”
[SIP]	IETF, RFC 3261, “SIP: Session Initiation Protocol”
[BCG]	ETSI, TS 102 539, “Digital Video Broadcasting (DVB);Carriage of Broadband Content Guide (BCG) information over Internet Protocol (IP)”
[TR069]	Broadband Forum, TR-069 Amendment 2, “CPE WAN Management Protocol v1.1”
[SIP-PRES]	IETF, RFC 3856, “A Presence Event Package for the Session Initiation Protocol (SIP)”
[TS183019]	ETSI, TS 183 019, “Network Attachment: User-Network protocol Interface Definitions”
[SIP-EVNT]	IETF, RFC 3265, “Session Initiation Protocol (SIP)-Specific Event Notification”
[TS183063]	ETSI, TS 183 063, “Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN);IMS-based IPTV stage 3 specification”
[TS102034]	ETSI, TS 102 034 V1.4.1 (2009-08), “Digital Video Broadcasting (DVB);Transport of MPEG-2 TS Based DVB Services over IP Based Networks”
[XCAP]	IETF, RFC 4825, “The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)”
[RFC3840]	IETF, RFC 3840, “Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)”
[RFC3455]	IETF, RFC 3455, “Private Header (P-Header) Extensions to the Session Initiation. Protocol (SIP) for the 3rd-Generation Partnership Project (3GPP)”
[SIP-IM]	IETF, RFC 3428, “Session Initiation Protocol (SIP) Extension for Instant Messaging”
[RTP]	IETF, RFC 3550, “RTP: A Transport Protocol for Real-Time Applications”

[TVA]	ETSI, TS 102 822-4, “Broadcast and On-line Services: Search, select, and rightful use of content on personal storage systems (“TV-Anytime”); Part 4: Phase 1 - Content referencing”
[SDP-TCP]	IETF, RFC 4145, “TCP-Based Media Transport in the Session Description Protocol (SDP)”
[SIP-REG]	IETF, RFC 3680, “A Session Initiation Protocol (SIP) Event Package for Registrations”
[RFC3803]	IETF, RFC 3803, “Content Duration MIME Header Definition”
[RFC3841]	IETF, RFC 3841, “Caller Preferences for the Session Initiation Protocol (SIP)”
[SMPL-IM]	OMA, Draft OMA-TS-SIMPLE_IM-V1_0-20080820-D, “Instant Messaging using SIMPLE”
[SMPL-PRES]	OMA, OMA-ERP-Presence_SIMPLE-V1_1-20080627-A, “Presence SIMPLE Specification”
[RTSP]	IETF, RFC 2326, “Real Time Streaming Protocol (RTSP)”
[RTSP2-AN]	IETF, draft-stiemerling-rtsp-announce-01, “RTSP 2.0 Asynchronous Notification”
[IGMP3]	IETF, RFC 3376, “Internet Group Management Protocol, Version 3”
[DLNA]	DLNA, “Networked Device Interoperability Guidelines”, October 2006
[GAA]	3GPP, TS 33.220, “Generic Authentication Architecture (GAA); Generic bootstrapping architecture”
[UB-UA]	3GPP, TS 24.109, “Bootstrapping interface (Ub) and network application function interface (Ua); Protocol details”
[ADDR]	IETF, RFC 1918, “Address Allocation for Private Internets”
[3G-SEC]	3GPP TS 33.203, “3G security; Access security for IP-based services”
[XCAP-EVT]	IETF, draft-ietf-sip-xcapevent-04, “An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Diff Event Package”
[XCAP-DFF]	IETF, draft-ietf-simple-xcap-diff-09, “An Extensible Markup Language (XML) Document Format for Indicating A Change in XML Configuration Access Protocol (XCAP) Resources”
[CEA2014A]	CEA, CEA-2014-A, “Web-based Protocol and Framework for Remote User Interface on UPnP Networks and the Internet (Web4CE)”, (including the August 2008 Errata).
[CLSLESS]	IETF, RFC 3442, “The Classless Static Route Option for Dynamic Host Configuration Protocol (DHCP) version 4”
[DHCP-OPT]	IETF, RFC 2132, “DHCP Options and BOOTP Vendor Extensions”
[DHCP-VND]	IETF, RFC 3925, “Vendor-Identifying Vendor Options for Dynamic Host Configuration Protocol version 4 (DHCPv4)”
[DOM-NAME]	IETF, RFC 1035, “Domain Names - Implementation And Specification”
[SDP-RTCP]	IETF, RFC 3556, “Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth”
[SES-TIMR]	IETF, RFC 4028, “Session Timers in the Session Initiation Protocol (SIP)”
[UPNP]	UPnP Forum, “UPnP Device Architecture”
[TS102472]	ETSI, TS 102 472 v1.2.1, “DVB IP Datacast”
[RTCP-XR]	IETF, RFC 3611, “RTP Control Protocol Extended Reports (RTCP XR)”
[PSS]	3GPP, TS 26.234v750, “Transparent end to end packet switched streaming service (PSS) – Protocols and Codecs”
[FEC]	IETF, RFC 4756, “Forward Error Correction Grouping Semantics in Session Description Protocol”
[SIP-CFG]	IETF, draft-ietf-sipping-config-framework-15 – “A Framework for Session Initiation Protocol User Agent Profile Delivery”
[RFC3994]	IETF, RFC 3994, “Indication of Message Composition for Instant Messaging”
[RFC3551]	IETF, RFC 3551, “RTP Profile for Audio and Video Conferences with Minimal Control”
[TR135]	Broadband Forum, TR-135, “Data Model for a TR-069 Enabled STB”

[TR106]	Broadband Forum, TR-106, “Data Model Template for TR-069 Enabled Devices”
[TR098]	Broadband Forum, TR-098, “Internet Gateway Device Data Model for TR-069”
[TR104]	Broadband Forum, TR-104, “DSLHome™ Provisioning Parameters for VoIP CPE”
[RFC3926]	IETF, RFC 3926, “FLUTE - File Delivery over Unidirectional Transport”
[SHA-1]	U.S. Department of Commerce/National Institute of Standards and Technology, FIPS PUB 180-1, “Secure Hash Standard”
[RFC4961]	IETF, RFC 4961, “Symmetric RTP/RTP Control Protocol (RTCP)”
[RFC4787]	IETF, RFC 4787, “Network Address Translation (NAT) Behavioral Requirements for Unicast UDP”
[ES282003]	ETSI, ES 282 003, “Resource and Admission Control Sub-system (RACS) - Functional architecture”
[SDP]	IETF, RFC 4566, “SDP: Session Description Protocol”
[RFC3986]	IETF, RFC 3986, “Uniform Resource Identifier (URI): Generic Syntax”

1.2 Open IPTV Forum References

[REQS]	Open IPTV Forum, “Service and Platform Requirements”, V1.1, July 2008.
[ARCH]	Open IPTV Forum, “Functional Architecture”, V1.2, January 2009.
[MEDIA]	Open IPTV Forum, “Release 1 Solution Specification, Volume 2 - Media Formats”, V1.2, August 2012.
[META]	Open IPTV Forum, “Release 1 Solution Specification, Volume 3 - Content Metadata”, V1.2, August 2012.
[DAE]	Open IPTV Forum, “Release 1 Solution Specification, Volume 5 - Declarative Application Environment”, V1.2, August 2012.
[CSP]	Open IPTV Forum, “Release 1 Solution Specification, Volume 7 - Authentication, Content Protection and Service Protection”, V1.2, August 2012.

1.3 Informative References

[ABNF]	IETF, RFC 4234, “Augmented BNF for Syntax Specifications: ABNF”

2 Conventions and Terminology

2.1 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 [RFC2119]. All sections and annexes, except “Introduction”, are normative, unless they are explicitly indicated to be informative.

2.2 Definitions

Term	***Definition***
Native HNI-IGI function (often shortened to Native HNI-IGI)	The procedures for interactions on the HNI-IGI interface are provided as part of the OITF implementation - typically in native code.
Non-native HNI-IGI function (often shortened to Non-native HNI-IGI)	The procedures for interactions on the HNI-IGI interface are provided by a service provider in JavaScript as part of a DAE application.

2.3 Abbreviations

All abbreviations used in this Volume are provided in Volume 1.

3 Interfaces

3.1 Consumer Network to Provider Network Interfaces (UNI)

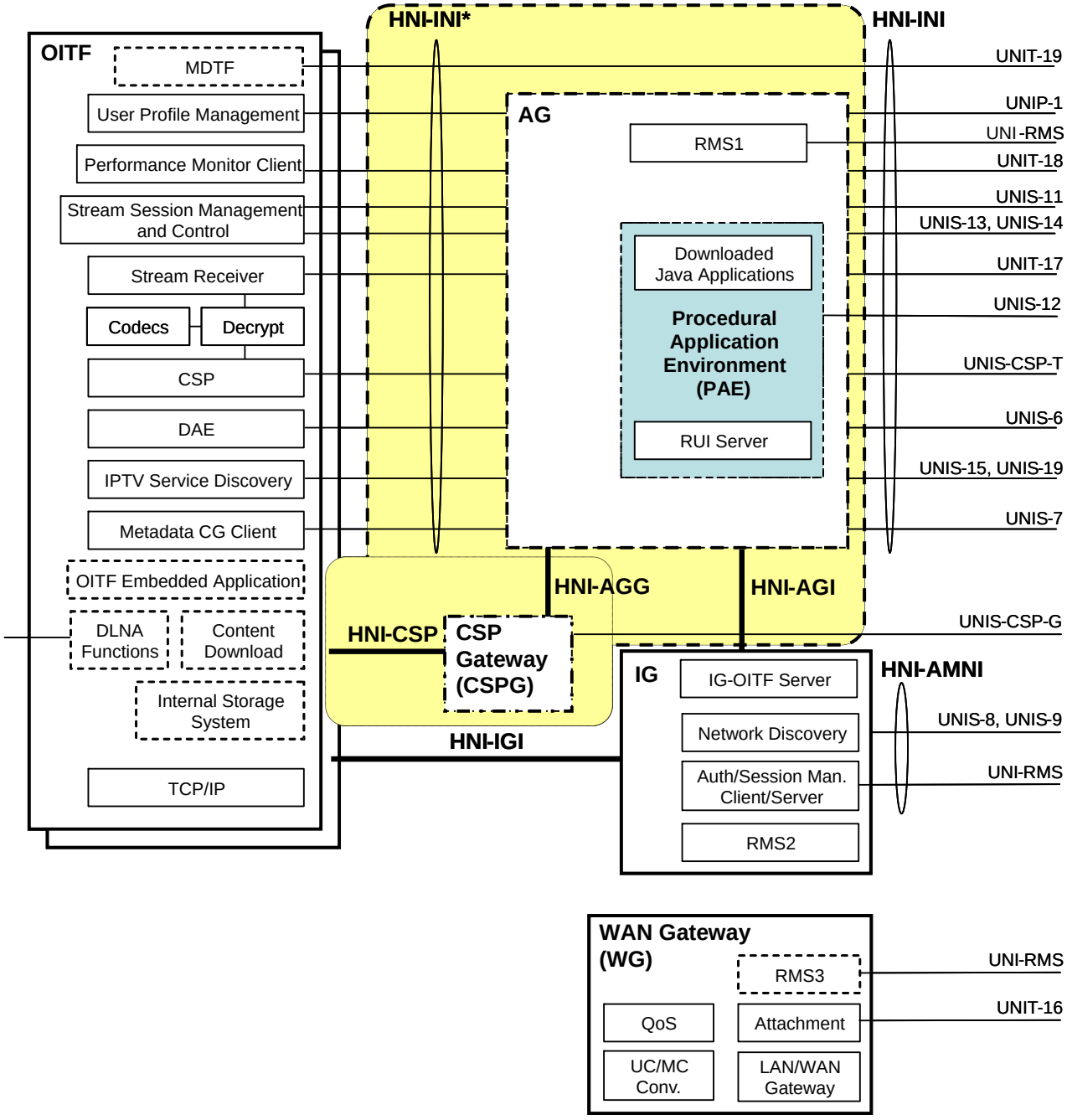


Figure 1: Consumer Network High Level Architecture

Figure 1 depicts the functional entities, functions and reference points defined by the Open IPTV Forum Functional Architecture **[ARCH]** in the Consumer Network.

In a device that implements both the OITF and the IG, the use of the HNI-IGI interface is OPTIONAL. In this case, the device SHALL support the UNIS-8, UNIS-9 and UNI-RMS interfaces.

The HNI-IGI interface consists of a set of interactions between the OITF and the IG. Certain interactions on the HNI-IGI interface MAY be implemented either natively or as a DAE application, whereas other interactions cannot be implemented as a DAE applications and MUST be implemented in native code. An OITF is said to implement the

"native HNI-IGI function" if it supports at least (but is not limited to) the interactions which MUST be implemented in native code. The case where no native interaction is supported is hereafter known as "non-native HNI-IGI function".

The interactions that must be implemented natively consist of user registration (sections 5.3.6.1 and 6.3.2.2) including service provider discovery (section 5.3.1.1), and GBA procedures (section 5.3.6.2) performed at OITF startup.

An OITF that supports the non-native HNI-IGI function can still be used in a managed network scenario, but without the support of GBA based authentication to application servers.

Table 1: UNI Reference Points and Protocols

Reference Point	***Description***	***Protocols***
UNIP-1	Reference point for user initiated IPTV service profile management.	HTTP, XCAP
UNIS-6	Reference point for user interaction with application logic for transfer of user requests and interactive feedback of user responses (provider specific GUI). HTTP and FLUTE is used to interface between the DAE and the IPTV Application Function in both the managed and unmanaged models.	HTTP, FLUTE
UNIS-7	Requests for transport and encoding of content guide metadata. The reference point includes the metadata and the protocols used to deliver the metadata, and SHALL be based on DVB-IP BCG. **[BCG]**	HTTP, DVBSTP
UNIS-8	Authentication and session management for managed network model.	IMS SIP
UNIS-9	Authentication for GBA Single-Sign on. See [CSP]	HTTP
UNIS-11	Reference point for control of real time streaming (e.g. control for pause, rewind, skip forward). The reference point includes content delivery session setup in case of unmanaged.	RTSP
UNIS-12	Reference point between the AG and the provider specific application functional entity.	HTTP, FLUTE
UNIS-13	User Stream control for multicast of real time content and data for the managed network model. This interface may also be used by service providers who wish to offer multicast services without session initiation.	IGMP
UNIS-14	Reference point used for authorization of service access for the unmanaged network model. See [CSP]	HTTP
UNIS-15	Reference point to the IPTV Service Discovery FE to obtain information about IPTV services offered by an IPTV Service Provider.	HTTP, DVBSTP
UNIT-16	Reference point used for Network Attachment.	DHCP
UNIT-17	Content stream including content; content encryption (for protected services) and content encoding. This reference point MAY be used for both multicast and unicast (UNIT-17M and UNIT-17U, respectively).	RTP, HTTP, UDP
UNIT-18	Performance monitoring interface for reporting the performance monitoring results.	RTCP, RTSP
UNIS-19	Reference point to the IPTV Service Provider Discovery functional entity to obtain the list of Service Providers, and related information.	HTTP
UNI-RMS	Remote Management using DSL Forum TR-069 framework [TR069]	HTTP/TR-069
UNIS-CSP-T	Rights management for protected content – including key management and rights expression. See [CSP]	HTTP/MARLIN

Table 2: Other interfaces

WAN gateway LAN Interfaces	Interface between OITF/IG and AG and the WAN Gateway	DHCP, IGMP

3.2 Provider Network Reference Points Description

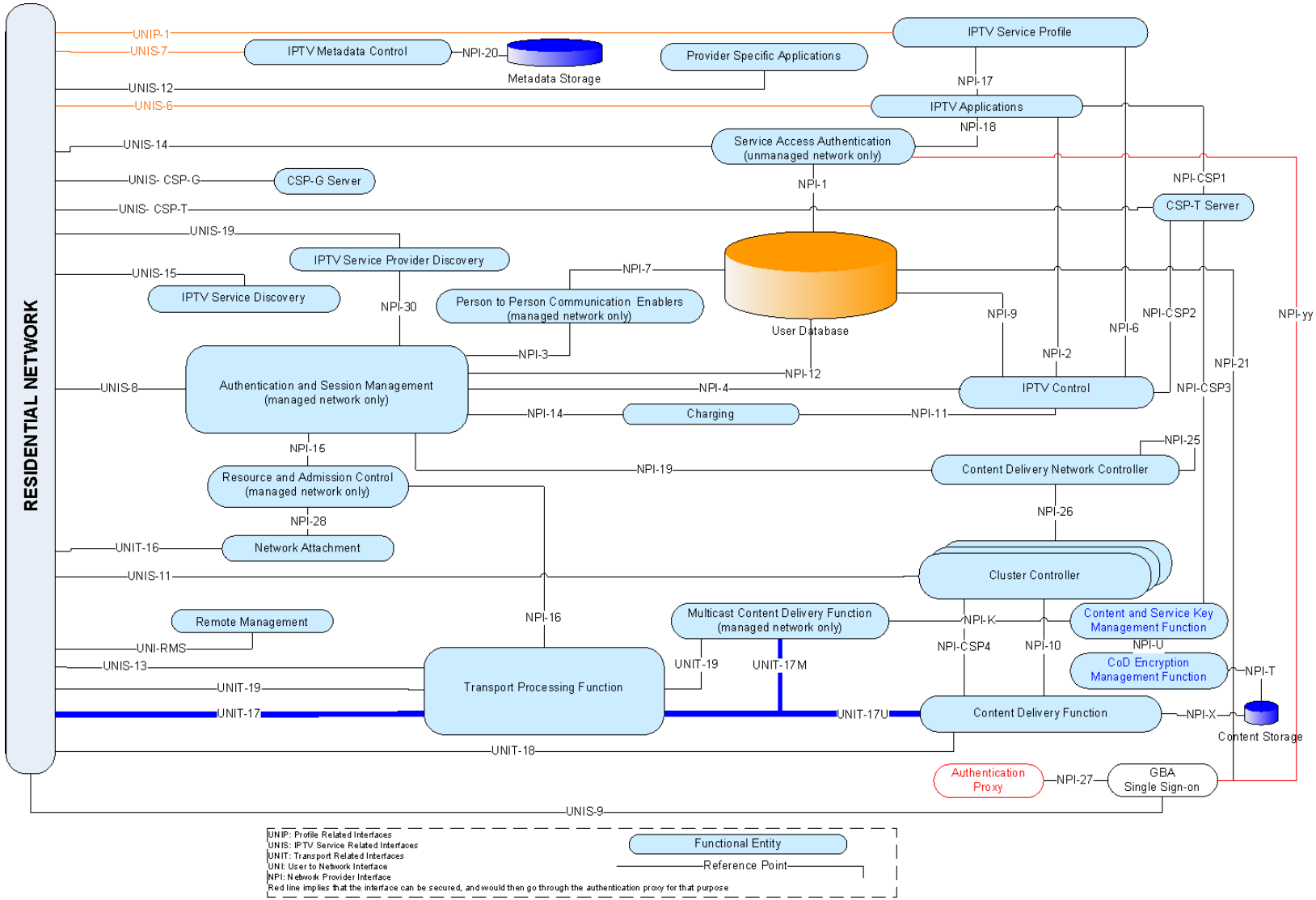


Figure 2: Provider Network High Level Architecture

Figure 2 shows the Functional Entities and Reference points in the service provider network defined in **[ARCH]**.

Table 3: NPI Reference Points and Protocols

Reference Point	***Description***	***Protocols***
NPI-1	Reference point between the Service Access Authentication FE and the User Database.	Not Specified
NPI-2	An OPTIONAL reference point allowing interaction between IPTV applications and the IPTV Control FE.	Not Specified
NPI-3	The reference point between Authentication Session Management and Person-to-Person Communication Enablers.	ISC interface [TS124503]
NPI-4	Reference point for routing of IPTV service related messages to the IPTV Control FE.	ISC interface [TS124503]
NPI-6	This reference point allows the IPTV Control FE to retrieve the subscriber's IPTV-related service data when a user registers in the IMS network.	Not Specified
NPI-7	This reference point allows person-to-person application enablers to retrieve the subscriber's IMS data from the user database.	Sh Interface [UMTS-SH]
NPI-9	This reference point allows the IPTV Control Point to retrieve the subscriber's IMS-specific data from the user database.	Sh Interface [UMTS-SH]
NPI-10	A reference point for the allocation/de-allocation and control of content for a unicast session.	RTSP
NPI-11	A reference point for sending events and charging information.	Rf and Ro [CHNG]
NPI-12	This reference point allows the Authentication and Session Management FE to retrieve the subscriber's IMS data from the User Database as a part of the user's IMS registration.	Cx [DIAM]
NPI-14	A reference point from Charging FE and Authentication and Session Management FE.	Rf [CHNG]
NPI-15	This reference point controls the Resources and Admission Control.	Gq' [AFSPDF]
NPI-16	Reference point between the Transport Processing Function and Resource and Admission Control.	Re [RACS-RE]
NPI-17	Reference point between the IPTV Applications and the IPTV Service Profile.	Not Specified
NPI-18	Reference point between the Service Access and Authentication FE and the IPTV Applications. This SHALL only be used in the unmanaged network model.	Not Specified
NPI-19	This reference point SHALL be used for unicast session setup control between the Authentication and Session Management and the Content Delivery Network Controller.	SIP/SDP
NPI-20	This OPTIONAL reference point allows the retrieval of CG data.	Not Specified
NPI-21	This reference point allows the GBA Single Sign-on functional entity to validate user credentials.	Not Specified
NPI-25	This reference point allows forwarding unicast control messages to the appropriate Content Delivery Network Controller FE.	SIP/SDP
NPI-26	The reference point allows the Content Delivery Network Controller to delegate the handling of a unicast session to a specific Cluster Controller.	SIP/SDP
NPI-27	The reference point between the Authentication Proxy and the GBA Single Sign-on node allows the proxy to retrieve a user key for authentication purposes.	Not Specified
NPI-28	This reference point SHALL be used to push the user access capabilities to the Network Attachment and the RAC.	e4 [NASS-E4]
NPI-30	This reference point supports the IPTV Service Provider Discovery step of the service discovery procedure for managed model.	ISC interface [TS124503]

3.3 Interfaces to External Systems

3.3.1 Consumer Network

Table 4: External Interfaces from the Consumer Network

DLNA Function	Interface between the OITF and DLNA devices in the home.	DLNA

4 Structure of the document

Each section of this specification identified below defines the procedures that use a specific protocol:-

Section 5: HTTP

Section 6: SIP and SIP/SDP

Section 7: RTSP

Section 8: IGMP and Multicast Protocol

Section 9: RTP/RTCP

Section 10: UPnP

Section 11: DLNA

Section 12: DHCP

Section 13: UDP

The annexes cover the following topics:

Annex A:

Annex B: Example of IPTV Protocol Sequences (informative)

Annex C: Example Messages (informative)

Annex D: User Profile Description (informative)

Annex E: Mapping attributes for Scheduled Content

Annex F: <protocol> names

Annex G: System Infrastructure

Annex H: System Infrastructure Mechanisms (informative)

Annex I: Presence XML Schema

Annex J: Protocol Procedure Section Structure (informative)

Annex K: OITF-specific TR-135 and TR-106 Remote Management Objects

Annex L: New Event package for SIP SUBSCRIBE /NOTIFY (informative)

Annex M: Schema Extension for FLUTE FDT

5 HTTP

5.1 HTTP Reference points

This section defines the protocol for the use of HTTP over the following reference points:

- HNI-IGI
  Certain interactions on the HNI-IGI interface can only be implemented natively, while the rest can be implemented either in native code or in a DAE application. In the following sections, if no qualification is provided it must be understood that the function can be performed natively or as a DAE application.
- UNIP-1
- UNIS-6
- UNIS-7
- UNIS-9
- UNIS-15
- UNIT-17
- UNIS-19

5.2 Protocol for IPTV Service Functions

5.2.1 Scheduled Content

5.2.1.1 Protocol over HNI-IGI for the Managed Model

When the OITF initiates, modifies or terminates a Scheduled Content service, the OITF sends HNI-IGI messages containing the appropriate method, mapped to HNI-IGI as described in section 5.5.1, “OITF-IG Interface (HNI-IGI)”.

The SIP-specific information in the related messages is described in section 6.2.1, “Scheduled Content Service.” The SIP-specific information is mapped to the HNI-IGI protocol, as described in section 5.5.1. In particular, the OITF creates HTTP headers for an HNI-IGI message by adding “X-OITF-” in front of the necessary SIP header names. In addition, optional parameters may be included as defined in **[TS124503]**.

Certain interactions on the HNI-IGI interface SHALL be implemented natively, while the remaining applicable interactions MAY be implemented either natively or as a DAE application. In the following sections, if no qualification is provided, it must be understood that the interaction can be performed natively or as a DAE application

5.2.1.1.1 Session Initiation

The HNI-IGI function in the OITF SHALL follow the following procedure for session initiation:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, “OITF-IG Interface (HNI-IGI).” The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - see Table 5
- **HTTP Request Body:** SDP offer containing the following elements (conforming to [TS183063]):
  - The m-line(s) SHALL be set to the Scheduled Content service which the OITF intends to join first, according to the mapping defined in section E.1, “Mapping SDP attributes from DVB SD&S information.”

- The c-line(s) SHALL be set to the Scheduled Content service which the OITF intends to join first, according to the mapping defined in section E.1, “Mapping SDP attributes from DVB SD&S information.”

- An a=bc_service: BCServiceId line SHALL indicate the Scheduled Content service that the OITF intends to join (according to the mapping defined in section E.1, “Mapping SDP attributes from DVB SD&S information”).

- Optionally one or more a=bc_service_package: <BCPackageId> as defined in section E.2, “Service Package SDP attributes.” The initial offer shall not contain mult_list and bc_tv_service_id_list parameter. If the initiation is the result of a previously denied initiation, the OITF may restrict the Scheduled Content services by including mult_list attributes.

- If the OITF has knowledge of the bandwidth of the Scheduled Content service with the highest bandwidth requirement included in the session, the b-line SHALL be included and set to this value. If the OITF supports FEC and the Scheduled Content service has FEC enabled, then the OITF SHALL include the additional bandwidth in the value set in the b=line. If the OITF does not support FEC and the Scheduled Content service includes FEC that uses the same multicast group address then the FEC bandwidth needs to be included.

- An a=recvonly line

In order for the OITF to connect to the FEC stream associated with the original multicast stream, additional parameters SHALL be included in the SDP offer as follows:

- An m-line for the FEC stream, as indicated by the Service Discovery or Metadata Control FE. The m-line SHALL be set according to the mapping defined in section E.1, “Mapping SDP attributes from DVB SD&S information.”

- A c-line according to the mapping defined in section E.1

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers for the process as per Table 5. The IG SHALL send a SIP INVITE to the network to request the initiation of the scheduled content session, and SHALL wait for the response to the request. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: On receipt of the response from the network the IG SHALL return a 200 OK HTTP response (or other appropriate received responses) to the OITF to report the response to the initiation request. The response SHALL include a list of SIP headers as per Table 6 in addition to the normal HTTP headers as per RFC 2616 [HTTP], and the same SDP answer body that was received by the IG in the SIP message.

Step 4: When the OITF receives the response to the INVITE, it SHALL examine the media parameters in the received SDP. The OITF SHALL restrict the Scheduled Content services that it joins according to the parameter (the a=bc_service_package attribute). received from the IPTV Control FE. However, if the OITF retrieved the IPTV user profile prior to session initiation, then it MAY ignore the=bc_service_package attribute.

If the OITF receives an error code with an Insufficient Bandwidth indication in the response from the IG, the OITF MAY perform a new INVITE with a reduced maximum bandwidth for the Scheduled Content service. This procedure MAY be repeated. If no agreement can be reached, the OITF MAY display a failure message to the user.

Step 5: Upon receipt of a 200 OK response, the OITF SHALL send an HTTP PENDING_IG to acknowledge the final response. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header**: Including the following:

  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**

  - <list of SIP headers encoded as HTTP headers> - as per Table 7

- **HTTP Request Body**: Empty

Table 5: Supported HTTP extension headers in the HNI-IGI INVITE Request message for Scheduled Content session setup (OITF→IG)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line The Request-URI in the INVITE request SHALL be the well known PSI (Public Service Identifier) of the Scheduled Content Service: OIPF_IPTV_SC_Service@<domain name>. The domain part SHALL be the IPTV Service Provider domain name obtained via Service Provider discovery.	RFC 3261 [SIP] INVITE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Contact The URI parameter MUST be included, and MUST match what is sent in the Contact header included in the registration request. The IG includes all other mandatory parameters that are absent.	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP] (application/sdp)
X-OITF-Content-Length	RFC 3261 [SIP]
X-OITF-Supported	RFC 3261 [SIP] set to timer
X-OITF-Session-Expires	RFC 4028 [SES-TIMR]

Table 6: Supported HTTP extension headers in the response message to an HNI-IGI INVITE request message for Scheduled Content session setup (IG→OITF)

X-OITF HTTP Headers	**Source of Information for Coding purposes**
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Session-Expires	RFC 4028 [SES-TIMR]
X-OITF-Content-Type	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]

Table 7: Supported HTTP extension headers in the HNI-IGI ACK message for a successful Scheduled Content session setup (OITF→IG)

X-OITF HTTP Headers	**Source of Information for Coding purposes**
X-OITF-Request-Line The Request-URI in the ACK request SHALL be the contact included in the response to the INVITE message	RFC 3261 [SIP] ACK <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line" of the initial request	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]

X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Contact The URI parameter MUST be included, and MUST match what has been inserted in the INVITE message. The IG includes all other mandatory parameters that are absent.	RFC 3261 [SIP]

5.2.1.1.2 Session Modification

To join a service outside the set of channels negotiated at session initiation, or to perform a bandwidth modification, the OITF SHALL send a request to the IG for session modification. The OITF SHALL generate a re-INVITE request, as defined in Table 5.

The OITF SHALL include an SDP offer in the session modification request. The format of this request SHALL be the same as for a session initiation.

5.2.1.1.3 Session Termination

To terminate a scheduled content session, the OITF SHALL use the following procedure:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 8
- **HTTP Request Body**: Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers needed for the message as per Table 8. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection. The IG SHALL send a SIP BYE to the network, to request the termination of the scheduled content session, and shall wait for the response.

Step 3: The IG SHALL then return a 200 OK HTTP response (or other appropriate responses) to the OITF to report the response to the Termination request. The response SHALL include, in addition to the normal HTTP headers as per RFC 2616 [HTTP], a list of SIP headers as per Table 9.

Table 8: Supported HTTP extension headers in HNI-IGI BYE Request for teardown of a Scheduled Content session (OITF→IG)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request- Line Note: The request URI MUST be set to the contact return in the 200 OK for the invite.	RFC 3261 [SIP] BYE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line" of the initial request	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

Table 9: Supported HTTP extension headers in the response to an HNI-IGI BYE Request for teardown of a Scheduled Content session (IG→OITF)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

5.2.1.1.4 Session Refresh

It is the responsibility of the OITF application to refresh the Scheduled Content session before the session expires. The IG SHALL consider a session terminated if it is not refreshed.

5.2.2 CoD

5.2.2.1 Protocol for session management for managed model over HNI-IGI

5.2.2.1.1 Retrieval of Session Parameters

If the OITF does not have all the necessary parameters to form the SDP offer the HNI-IGI function in the OITF SHALL retrieve missing SDP parameters using the following procedure:

Step 1: The OITF SHALL send an HTTP POST request to the IG on the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 10
- **HTTP Request Body**: Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers required for the outgoing message as per Table 10. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response that includes the reason for rejection.

Step 3: The IG SHALL send a SIP OPTIONS message to the network, to retrieve missing SDP parameters and SHALL wait for the response to the request. The IG SHALL then return a 200 OK HTTP response (or other appropriate responses) to the OITF to report the response to the request for missing SDP parameters. The response includes a list of SIP headers as per Table 11, in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**, as well as an SDP body containing the missing SDP parameters according to section 6.2.2.1.2, "Protocol over NPI-4, NPI-19, NPI-26."

Table 10: Supported HTTP extension headers in HNI-IGI OPTION Request for CoD session setup parameters (OITF→IG)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line Note: The request URI SHALL be set to the PSI (Public Service Identifier) of the CoD Services as follows: OIPF_IPTV_CoD_Service_*@<domain name> Where: - The wild card part (*) is a content instance identifier, constructed according to clause 4.3.2 in **[META]** when CoD content identifiers are delivered via the Content Guide. For DAE applications signalling CoD, the wild card part is constructed according to clause 8.1.2 in **[DAE]**. - The domain part (<domain name>) is the IPTV Service Provider domain name, obtained from the IPTV Service Provider discovery function.	RFC 3261 [SIP] OPTIONS <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To SHALL be set to the value of the request URI in the "X-OITF-Request-Line OPTIONS" header	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Accept	Set to application/sdp as per RFC 3261 [SIP]

Table 11: Supported HTTP extension headers in the response to an HNI-IGI OPTION Request for CoD session setup parameters

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]

5.2.2.1.2 Session Initiation

The OITF SHALL initiate the request for a Content on Demand session using the following procedure.

Step 1: The OITF SHALL send an HTTP POST request to the IG on the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 12
- **HTTP Request Body**: The request body includes the SDP offer generated by the OITF. The SDP offer SHALL include a media description for the RTSP content control channel and the media description for the content delivery channel. SDP shall be used as specified in **[TS124503]**.

- **SDP Parameters for the RTSP control channel**

  The RTSP content control media description SHALL be carried by TCP and follow **[SDP-TCP]**. Hence, the SDP parameters for the RTSP content control channel SHALL be set as follows:-

  - An m-line for an RTSP stream of format: m=<media> <port> <transport> <fmt>
    - The <media> field SHALL have a value of "application".
    - The <port> field SHALL be set according to **[SDP-TCP]**. The "a=setup" attribute SHALL be set to 'active', and port field SHALL be set to a value of "9", which is the discard port.
    - The <transport> field SHALL be set to "TCP" or "TCP/TLS". The former SHALL be used when RTSP runs directly on top of TCP and the latter SHALL be used when RTSP runs on top of TLS, which in turn runs on top of TCP.
    - The <fmt> parameter SHALL be included and set to "iptv_rtsp" (ex. `m=application 9 tcp iptv_rtsp`)
  - An "a=setup" attribute SHALL be present and set to "active" as defined in **[SDP-TCP]** (ex. `a=setup:active`)
  - An "a= connection" attribute SHALL be present and set as "new" as defined in **[SDP-TCP]** (ex. `a=connection:new`)
  - A c-line SHALL include the network type with the value set to "IN", the address type set to "IP4" and IP address of the RTSP content control stream. (ex. `c=IN IP4 <IP_ADDRESS>`)
  - One or more a=fmtp lines representing RTSP specific attributes set as follows:
    - The RTSP Version MAY be specified in a "fmtp:iptv_rtsp version" parameter.

- **SDP Parameters for the content delivery channels**

  For each media stream controlled by the RTSP content control channel the SDP offer SHALL include a content delivery channel media description, set as follows:

  - The m-line indicates the type of the media ("video"), the transport protocol and the port of the related content delivery channel as follows: m=<media> <port> <proto> <fmt>
    - <fmt> settings:
      - When MPEG2-Transport Stream [MPEG2-TS] is used, <fmt> SHALL be "33" as specificed in RFC 3551 **[RFC3551]**
      - When optional Timestamped-TS defined by **[DLNA]** is used, the RTP/AVP dynamic payload type SHALL be used and <encoding name> of "a=rtpmap" line SHALL be "vnd.dlna.mpeg-tts" as specified in **[DLNA]**, e.g.

        m=video 49232 RTP/AVP 98
        a=rtpmap:98 vnd.dlna.mpeg-tts/27000000

    - <proto> settings:
      - <proto> SHALL be set according to information obtained by the OITF either by OPTIONS or in the service access stage. If streaming is RTP, <proto> SHALL be set to RTP/AVP. If streaming is direct over UDP, <proto> SHALL be set to "MP2T/H2221/UDP" or "RAW/RAW/UDP"
  - The "c- line" SHALL include the network type with the value set to "IN", the address type set to "IP4" followed by the address of the OITF. (e.g, `c=IN IP4 <IP_ADDRESS>`)
  - The "b-line" SHALL contain the proposed bandwidth obtained by the OITF either by OPTIONS or during the service access phase. If the media stream is FEC protected and the OITF wishes to use one or more FEC streams, the bandwidth SHALL be the sum of the media stream bandwidth and the bandwidths of all the FEC stream to be used by the OITF. If the OITF cannot obtain the bandwidth, the

b= attribute SHALL be set to a pre-configured value.
(e.g,. `b=AS:15000`)

- A b=RR:<bandwidth-value>, line indicating the bandwidth value (in kbps) that the OITF proposes to use for sending Receiver Reports (RR). If this value is set to zero by the OITF, then it means that the OITF can not, or does not, wish to send Receiver Reports. This is the default setting, as explained in section 9.1.2.1, "Protocol over UNIT-17." Note that if the OITF sends RTCP Receiver Reports, then these can be used as keep-alive messages, as shown in section 6.2.2.2.5, "Protocol over NPI-26."

- An "a=" line with a "recvonly"
(e.g, `a=recvonly`)

If a media stream is FEC protected, the OITF MAY include the following for each FEC protected stream:

- One or more m-line for the FEC streams indicated in the response to the OPTIONS request. The m-lines shall be set according to the returned response.

In case there are multiple media streams to be FEC protected, or a single media stream protected by multiple FEC streams, grouping line(s) SHALL be included for the purpose of associating FEC stream(s) with media stream(s), one for each media stream m-line that is associated to a FEC stream. The grouping line uses the "FEC" semantic as defined in RFC 4756 **[FEC]**:

- a=group:FEC:<original stream id> <base FEC stream id> <enhancement FEC stream id>

The original stream id SHALL reflect the value held by the media description of media stream in the a=mid attribute. This implies that, when a grouping line is included, there SHALL be an additional media identification attribute within the m-line of the original media stream that is within the grouping line. The format for that attribute is:

- a=mid:<original stream id>

The base FEC stream id SHALL reflect the value held by the media description of the FEC stream (associated to the original stream) in the a=mid attribute.

Only base FEC stream SHALL be supported in Open IPTV Forum Release 1.

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers needed for the outgoing message, as per Table 12. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL send a SIP INVITE to the network, to request the initiation of a unicast session, and SHALL wait for the response to the request. The IG SHALL then return a 200 OK HTTP response (or other appropriate responses) to the OITF to report the response to the initiation request. The response includes a list of SIP headers as per Table 13, in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**, and the same SDP answer body received by the IG, as described in section 6.2.2.2.5, "Protocol over NPI-26."

Step 4: Upon receipt of a 200 OK response, the OITF SHALL send an HTTP Pending Request to acknowledge the final response. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 14
- **HTTP Request Body:** Empty

When parsing the b=RR:<bandwidth-value> line by the OITF: if the bandwidth value agreed is non-zero, then the OITF SHALL send RTCP RRs and SHALL NOT send RTSP keep-alive messages. If the bandwidth value received is zero, then the OITF SHALL NOT send RTCP RRs but instead it SHALL send RTSP keep-alive messages.

Table 12: Supported HTTP extension headers in HNI-IGI INVITE Request for CoD session setup (OITF→IG)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line Note: The request URI MUST be set to the PSI (Public Service Identifier) of the CoD Services as follows: IPTV_CoD_Service_*@<domain name> Where: - The wild card part (*) is a content instance identifier, constructed according to clause 4.3.2 in **[META]** when CoD content identifiers are delivered via the Content Guide. For DAE applications signalling CoD, the wild card part is constructed according to clause 8.1.2 in **[DAE]**. - The domain part (<domain name>) is the IPTV Service Provider domain name, obtained from the IPTV Service Provider discovery function.	RFC 3261 [SIP] INVITE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To MUST be set to the value of the request URI in the "X-OITF-Request-Line INVITE" header	RFC 3261 [SIP]
X-OITF-Contact Notes: URI parameter SHALL be included and SHALL match what is sent in the contact header included in the registration request. Expires parameter SHOULD be included.	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Supported	RFC 3261 [SIP] set to timer
X-OITF-Session-Expires	RFC 4028 [SES-TIMR]

Table 13: Supported HTTP extension headers in the response to an HNI-IGI INVITE Request for CoD session setup (IG→OITF)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]
X-OITF-Session-Expires	RFC 4028 [SES-TIMR]

Table 14: Supported HTTP extension headers in HNI-IGI ACK Request for successful CoD session teardown (OITF→IG)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line The Request-URI in the ACK request SHALL be the contact included in the response to the INVITE message	RFC 3261 [SIP] ACK <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]

X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line" of the initial request	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Contact The URI parameter SHALL be included, and SHALL match what as been inserted in the INVITE message. The IG includes all other mandatory parameters that are absent.	RFC 3261 [SIP]

5.2.2.1.3 Session Termination

The OITF SHALL send the request for a Content on Demand session termination using the following procedure:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 15
- **HTTP Request Body:** Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers required for the outgoing message, as per Table 15. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL send a SIP BYE to the network, to request the termination of a unicast session, and shall wait for the response. The IG SHALL then return a 200 OK HTTP response (or other appropriate responses) to the OITF to report the response to the termination request. The response includes a list of SIP headers as per Table 16 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Table 15: Supported HTTP extension headers in HNI-IGI BYE Request for CoD session teardown (OITF→IG)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line	RFC 3261 [SIP] BYE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line" of the initial request	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

Table 16: Supported HTTP extension headers in the response to an HNI-IGI BYE Request for CoD session teardown (IG→OITF)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]

X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

5.2.2.1.4 Session Refresh

It is the responsibility of the OITF application to refresh the CoD session before the session expires. The IG SHALL consider a session terminated if it is not refreshed

5.2.2.2 Protocol for streaming for unmanaged model over UNIT-17

The use of the HTTP protocol on this reference point SHALL comply with **[HTTP]**

The Content Delivery Function SHALL support the Range HTTP header in a GET request from the OITF to reduce unnecessary network usage by allowing partial retrieval for use in cases such as trick play. The OITF MAY pre-buffer the content in order to sustain play-out even when the HTTP transfer is stalled.

5.2.3 Content Download

Content Download is a service where IPTV content is downloaded to the optional Internal Storage System in the OITF. The OITF may play-out the content while downloading. Trick play MAY be performed within the downloaded content depending on the content rights.

5.2.3.1 Protocol over UNIT-17

The use of the HTTP protocol on this reference point SHALL comply with **[HTTP]**

The Content Delivery Function SHALL support the Range HTTP header in a GET request from the OITF to reduce unnecessary network usage by allowing partial retrieval.

5.3 Protocol for Service Access and Control Functions

5.3.1 Service Provider Discovery

5.3.1.1 Protocol over HNI-IGI for the Managed Model

5.3.1.1.1 Retrieval of Service Provider Discovery Information

The procedures in this section SHALL only be performed in the context of the default user. When the OITF supports native HNI-IGI, it SHALL follow the following procedure to retrieve Service Provider Discovery Information:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, “OITF-IG Interface (HNI-IGI).” The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 17
- **HTTP Request Body:** Empty or optionally, the OITF MAY include a body associated with the appid “urn:oipf:application:iptv-SP-discovery”.

  The optional message body sent to the Service Provider Discovery FE SHALL include the capabilities of the OITF. The Content-Type of the message body SHALL be set to “application/vnd.oipf.ueprofile+xml”, which refers to the MIME type of the schema defined in section D.2. See Table 17 for X-OITF-Content-Type header.

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers required for the outgoing message as per Table 17. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: If the IG has the requested information, it SHALL respond immediately with HTTP 200 OK. If not, the IG SHALL send a SIP SUBSCRIBE to the network, to subscribe to the "ua-profile" event, and SHALL wait for the response to the subscription request. The IG SHALL then return a 200 OK HTTP response (or other appropriate responses) to the OITF to report the response to the subscription request. The response includes a list of SIP headers as per Table 18 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Step 4: The OITF SHALL send an HTTP HNI-IGI PENDING_IG request (refer to section 5.5.1.1, "HNI-IGI Message Types"), and SHALL wait for any incoming messages.

Step 5: When a SIP NOTIFY is received by the IG for a "ua-profile" event, the IG SHALL return a HTTP 200 OK response to the OITF. The response includes a list of SIP headers as per Table 19 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**. The body of the HTTP response SHALL be the SIP body received in the incoming NOTIFY message. The content of the HTTP Response SHALL be as follows:

- **HTTP Response Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 19
- **HTTP Response Body:** Body of the incoming NOTIFY

The OITF SHALL parse the XML document in the body to ensure that it complies with the schema defined in section 3.2.1 of **[META]**.

When parsing the list of parameters, the OITF SHALL take the following action:

- If the Service Provider Discovery Information for a Service Provider is already present in the OITF (i.e., for which the OITF already has an entry), and
  - If the "@Version" attribute does not have the same value as that received in the NOTIFY message, then the OITF SHALL perform the following actions:
    - The OITF SHALL update its parameters with the new values sent by the Service Provider Discovery FE. Also if the Segment@ID or Segment@Version has changed, the OITF SHALL update the service discovery information with that received from the Service Discovery FE.
  - If the "@Version" attribute has the same value as that received in the NOTIFY message, the OITF SHALL NOT update the stored Service Provider Discovery information.
- If the Service Provider Discovery Information for a Service Provider is not known to the OITF (i.e., the OITF does not have an entry for the Service Provider Discovery Information)
  - The OITF SHALL create a new entry for the new Service Provider with all the parameters received in the NOTIFY message.

The IPTV Service Provider Discovery Information delivered via this protocol SHALL conform to ETSI TS 102 034 **[TS102034]** section 5.2.5, with the extended element defined in the Metadata Specification **[META]**.

Step 6: Once the OITF accepts the HTTP message containing the incoming SIP NOTIFY, it SHALL send an HTTP HNI-IGI PENDING_IG request to the IG. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 20
- **HTTP Request Body:** Empty

Step 7: The IG SHALL send the SIP 200 OK response to the network and then SHALL return to Step 5 to handle any subsequent NOTIFY received from the network.

5.3.1.1.2 Procedure for Cancellation of the Subscription

The procedure for de-registering the IPTV default user MUST be preceded with a cancellation of subscription.

The procedure is the same as the procedure for initiating a subscription to the "ua-profile", except that the X-OITF-Expires header in Table 17 SHALL be set to 0.

Table 17: Supported HTTP extension headers in HNI-IGI SUBSCRIBE Request for SP Discovery

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line Note: The request URI SHALL be set to the well known PSI. The PSI SHALL be composed of the domain name extracted from the public user identity with a user part set to "OIPF_IPTV_SPD". (e.g., OITF_IPTV_SPD@<domain_name>)	RFC 3261 [SIP] SUBSCRIBE <Request URI> SIP/2.0
X-OITF-From Note: The From user MUST be set to the IMPU of the default user.	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Event Extend the existing "ua–profile" event package for SIP SUBSCRIBE request The Event header SHALL be set to the "ua-profile" event package. The Event parameters SHALL be set as follows: - The "profile-type" parameter SHALL be set to "application". - The "appids" parameter SHALL be set to "urn:oipf:application:iptv-SP-discovery".	RFC 3265 [SIP-EVNT] and as per ETSI 183 063 [TS183063] section 5.1.2.2.1
X-OITF-Contact Notes: 1. URI parameter MUST be included, and MUST match the value that is sent in the Contact header in the registration request. 2. Expires parameter SHOULD be included 3. Priority parameter SHOULD be included The IG includes all other mandatory parameters that are absent.	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Expires Note: If absent a default value according to RFC 3261 [SIP] SHALL be assumed by the IG To cancel the subscription, the X-OITF-Expires SHALL be set to 0	RFC 3261 [SIP]
X-OITF-Accept Set to "application/vnd.oipf.spdiscovery+xml"	RFC 3261 [SIP]
X-OITF-Content-Type Included, optionally, when signalling OITF capabilities according schema defined in section D.2. It SHALL be set to "application/vnd.oipf.ueprofile+xml"	RFC 3261 [SIP]

Table 18: Supported HTTP extension headers in the response to an HNI-IGI SUBSCRIBE Request for SP Discovery

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Expires	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]

Table 19: Supported HTTP extension headers in the NOTIFY request to the SUBSCRIBE to SP discovery

X-OITF HTTP Headers	Source of Coding Information
X-OITF-Request-Line Note: The Request URI MUST match the contact URI included in the contact field of the SIP SUBSCRIBE	RFC 3261 [SIP], RFC 3265 **[SIP-EVNT]** and draft-ietf-sipping-config-framework-15 **[SIP-CFG]** NOTIFY <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Event	RFC 3265 [SIP-EVNT] and as per ETSI 183 063 [TS183063] section 5.2.2.2
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-Subscription-State	RFC 3265 [SIP-EVNT] and RFC 3856 [SIP-PRES]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type Set to “application/vnd.oipf.spdiscovery+xml”	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]

Table 20: Supported HTTP extension headers in the response to a NOTIFY request to the SUBSCRIBE to SP discovery

XOITF HTTP Headers	Source of Coding Information
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]

Note: Cancellation of subscription is not required if the X-OITF-Expires header was set to 0 in the initial SUBSCRIBE request.

5.3.1.1.3 Refreshing the Subscription

The procedure for refreshing a subscription is the same as the procedure for initiating a subscription

The application initiating the subscription procedure SHALL refresh the subscription based on the refresh subscription timer information received in the response to the subscription. Refreshing a subscription SHOULD be performed before the expiry of the refresh timer. A subscription that is not refreshed will be terminated.

The IG SHALL consider a subscription terminated if is not refreshed.

5.3.1.2 Protocol over UNIS-19 for the Unmanaged Model and Non-native HNI-IGI

The OITF retrieves the Service Provider Discovery entry point and uses the entry point to retrieve a list of IPTV service providers using HTTP for that purpose. The IPTV Service Providers list SHALL be delivered as SD&S records or DAE applications.

When an IPTV service provider discovery entry point is selected, Service Provider Discovery information SHALL be delivered as Service Discovery and Selection (SD&S) records or as DAE applications. This information is provided by the Service Platform Provider.

When SD&S records are used, the HTTP protocol conforming to ETSI TS 102 034 **[TS102034]** section 5.4.2 SHALL be used for the transport of IPTV Service Provider Discovery Information. The data delivered SHALL conform to ETSI TS 102 034 **[TS102034]** section 5.2.5, with the extension defined in **[META]**.

When DAE applications are used, the HTTP protocol and data formats SHALL conform to section 5.2.2 of Open IPTV Forum Solution Specification Volume 5 - Declarative Application Environment [DAE].

5.3.2 Service Discovery

5.3.2.1 Protocol over UNIS-6

The protocol on UNIS-6 SHALL be HTTP as defined in [DAE] for DAE application based service discovery. This protocol is used for the unicast transport of HTML ECMAScript documents between the OITF DAE function and the IPTV Application Functional Entity.

5.3.2.2 Protocol over UNIS-15

The protocol used on UNIS-15 for the transport IPTV Service Discovery information SHALL be HTTP conforming to ETSI TS 102 034 **[TS102034]** section 5.4.2.

The IPTV Service Discovery information delivered via this protocol SHALL conform to ETSI TS 102 034 **[TS102034]** section 5.2.6 with the extension defined in **[META]**

5.3.3 Service Access

5.3.3.1 Protocol over UNIS-6

UNIS-6 MAY be used for the unicast transport of HTML ECMAScript documents between the OITF DAE function and the IPTV Application functional entity for DAE application based service access.

See **[DAE]** for the details of the document format delivered via this protocol.

5.3.3.2 Protocol over UNIS-7

The use of the HTTP protocol on this reference point SHALL comply with section 4.1.2.2.2 (container based delivery) or section 4.2 (query mechanism) of the DVB-IP Broadband Content Guide specification **[BCG]**.

The Content Guide metadata delivered via this protocol SHALL conform to ETSI TS 102 539 **[BCG]** with the extension defined in **[META]**

The OITF MAY request user specific information from the Metadata Control FE based on the IPTV Subscription Profile. (See section 5.3.4, “Subscription profile management and usage.”)

5.3.4 Subscription profile management and usage

5.3.4.1 Protocols on UNIP-1

The OITF SHALL be able to obtain a user’s IPTV Subscription Profile. See section B.2.2, “IPTV Service Profile Manipulation through XCAP.” The format of the IPTV Subscription Profile SHALL conform to section D.1, “IPTV Subscription Profile.” The IPTV Subscription Profile MAY be used for filtering the Broadband Content Guide metadata, i.e. for the provision of a personalised content guide.

The IPTV Service Profile Functional Entity SHALL expose XCAP Server behaviour (HTTP Server 1.1, XML parser, and data repository) as defined in **[XCAP]**.

UNIP-1 SHALL comply with XCAP as defined in RFC 4825 [XCAP].

5.3.4.1.1 XCAP Application Usage for IPTV Service

Profile Management

The XML Configuration Access Protocol (XCAP) defined in RFC 4825 **[XCAP]** is used for manipulating data stored in the IPTV Service Profile Functional Entity. XCAP allows a client to read, write and modify application configuration data, stored in XML format, on a server. XCAP maps XML document sub-trees and element attributes to HTTP URIs, so that these components can be directly accessed by HTTP. XCAP uses the HTTP methods PUT, GET, and DELETE to operate on documents stored in the Service Profile Functional Entity.

The data stored in the IPTV Service Profile Functional Entity relates to the operation of the IPTV service. This specification defines a new Application Usage to allow a client to manipulate data related to IPTV services.

XCAP requires the definition of XML documents that are compliant with the XML schema and constraints defined for a particular XCAP application usage. The application usage defines the XML schema for the data used by the application, along with other key pieces of information.

Central to XCAP is the construction of the HTTP URI that points to a particular document or certain components of it. A component in an XML document can be an XML element, attribute, or the value of it.

XCAP application usage

XCAP requires application usages to fulfil a number of steps in the definition of such application usage. The remainder of this section specifies the required definitions of the IPTV services XCAP Application Usage.

Application Unique ID (AUID): Each XCAP application usage is associated with a unique name called the Application Unique ID (AUID). The AUID defined by this application usage falls into the vendor-proprietary namespace of XCAP AUID, where Open IPTV Forum is considered a vendor.

The proposed AUID to be allocated to the Open IPTV Forum IPTV services application usage SHALL be

org.openiptvforum.iptv

XML schema: Implementations in compliance with this specification SHALL implement the XML schema defined in Annex D.

Default namespace: XCAP requires application usages to declare the default namespace. The default namespace of the IPTV services XCAP application usage SHALL be

urn:oipf:params:xml:ns:iptv

MIME Type: The MIME type of IPTV service XML document SHALL be

application/vnd.oipf.userprofile+xml

Validation constraints: This specification does not specify any additional constraints beyond those defined by XCAP.

Data Semantics: The XML schema does not accept URIs that could be expressed as a relative URI reference causing a resolution problem. However, each of the supplementary services should consider if relative URIs are allowed in the subdocument tree, and in that case, they should indicate how to resolve relative URI references. In the absence of further indications, relative URI references should be resolved using the document URI as the base of the relative URI reference.

Naming conventions: By default, IPTV Service Profile XML documents are stored in the IPTV Service Profile Functional Entity. In order to facilitate the manipulation of an IPTV Service Profile XML document, the default XML file name SHALL be:

iptvprofile.xml

Resource interdependencies: This specification does not specify additional resource interdependency beyond those specified in the XML schema.

Authorization policies: The authorization policy for access and manipulation of an IPTV Service Profile document SHALL be defined by the Service Provider.

5.3.4.2 Protocols over HNI-IGI

5.3.4.2.1 Subscription to notification of changes in the IPTV Service Profile

The procedure for subscription to notification of changes in the IPTV service profile SHALL be invoked from either a DAE application or an embedded application in the OITF. The procedures SHALL be as follows:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in 5.5.1 OITF – IG Interface (HNI-IGI). The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 21.
- **HTTP Request Body:** The body contains the list of the requested URIs associated with the XCAP resources for which the subscription is issued. The MIME Type of the document inserted in the body will be signalled by the Content-Type header, set to "application/vnd.oipf.userprofile+xml".

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers needed for the outgoing subscription message, as per Table 21. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL send a SIP SUBSCRIBE to the network, to subscribe to the "xcap-diff" event package, and shall wait for the response to the subscription request. The IG SHALL return a HTTP 200 OK response to the OITF to report the response to the subscription request. The response SHALL include a list of SIP Headers as per Table 22 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Step 4: The OITF SHALL send an HTTP HNI-IGI PENDING_IG request (refer to section 5.5.1.1, "HNI-IGI Message Types"), and SHALL wait for any incoming messages.

Step 5: When a SIP NOTIFY is received by the IG, the IG SHALL return a HTTP 200 OK response to the OITF that includes the information carried in the incoming NOTIFY. The response SHALL include a list of SIP headers as per Table 23 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**. The body of the HTTP response SHALL include the "xcap-diff+xml" document carried in the NOTIFY body. This document contains the changes in the XCAP document(s) identified in the subscription request in Step 1(b).

Step 6: When the OITF accepts the incoming SIP NOTIFY, it SHALL send an HTTP POST PENDING_IG request to the IG to acknowledge the receipt of notification. The content of the HTTP request SHALL be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 24.
- **HTTP Request Body:** Empty

Step 7: The IG SHALL send the SIP 200 OK response to the network and then SHALL return to Step 5 to handle any subsequent NOTIFY messages that may be received from the network.

Table 21: Supported HTTP extension headers in HNI-IGI SUBSCRIBE Request for receiving notification of changes in the IPTV Service Profile

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line Note: The request URI MUST be set to the well known PSI of the IPTV Service Profile FE: The PSI SHALL be "OIPF_IPTV-ServiceProfile@<domainname>" where <domainname> SHALL be the IPTV Service Provider domain name obtained through Service Provider discovery.	RFC 3261 [SIP], RFC 3265 **[SIP-EVNT]** and draft-ietf-sip-xcapevent-03 **[XCAP-EVT]** SUBSCRIBE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Event The Event header SHALL be set to the "xcap-diff" event package.	RFC 3265 [SIP-EVNT] and as per ETSI 183 063 [TS183063] section 5.1.5.1
X-OITF-Accept The Accept header shall include the value "application/xcap-diff+xml". This header indicates the body formats allowed in subsequent NOTIFY requests	RFC 3265 [SIP-EVNT] and as per ETSI 183 063 [TS183063] section 5.1.5.1.
X-OITF-Content-type SHALL be set to "application/vnd.oipf.userprofile+xml" as the MIME Type of IPTV Subscription Profile schema.	RFC 3265 [SIP-EVNT]
X-OITF-Contact Notes: The URI parameter SHALL be included and SHALL match what is sent in the Contact header included in the registration request The Expires parameter SHOULD be included	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Expires Notes: If absent a default value shall be assumed by the IG	RFC 3261 [SIP]

Table 22: Supported HTTP extension headers in the response to an HNI-IGI SUBSCRIBE Request

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Expires	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]

Table 23: Supported HTTP extension headers in the NOTIFY request containing changes in the IPTV Service Profile

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Notes: The Request URI MUST match the contact URI included in the contact field of the SIP SUBSCRIBE	RFC 3261 [SIP] NOTIFY <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]

X-OITF-To	RFC 3261 [SIP]
X-OITF-Event	RFC 3265 [SIP-EVNT] and as per ETSI 183 063 [TS183063] section 5.1.5.2
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-Subscription-State	RFC 3265 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type The content-type header SHALL be set to "application/xcap-diff+xml".	RFC 3265 [SIP-EVNT] and as per ETSI 183 063 [TS183063] section 5.1.5.2

Table 24: Supported HTTP extension headers in the response to a NOTIFY request

X-OITF HTTP Header	Source of Coding Information
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]

5.3.4.2.2 Refreshing the Subscription

It is the responsibility of the application initiating the subscription procedure to refresh the subscription according to the "refresh subscription timer" parameter received in the response to the subscription request. Refreshing the subscription SHOULD be performed before the expiry of the refresh timer. A subscription that is not refreshed SHALL be terminated after the expiration of the timer.

The IG SHALL consider a subscription terminated if is not refreshed

5.3.4.2.3 Procedure for Cancellation of a Subscription

This procedure MAY be invoked at any time.

The procedure for de-registering the IPTV end user SHALL be preceded by the cancellation of any subscription for notification of changes in the user's IPTV Service Profile.

The procedure for cancellation of the subscription is the same as the procedure for initiating a subscription to the ua-profile event package, except that the X-OITF-Expires header in Table 21 SHALL be set to 0.

5.3.5 Remote Management

5.3.5.1 General Procedures on UNI-RMS

The remote management functions required for managed devices are specified in the general framework document TR069 **[TR069]** by the Broadband Forum. The framework document is associated with a number of Technical Reports that define the **CWMP data models** that are specific for each device function.

5.3.5.1.1 UNI-RMS for IG, AG and WAN Gateway

In addition to TR-069, the following specifications SHALL apply:

- TR-098 **[TR098]** that defines the data model for the "internet gateway device" SHALL apply to the WAN Gateway FE (see RMS3 functional block of the "Open IPTV Forum – Functional Architecture" document **[ARCH]**.
- TR-106 **[TR106]** that defines the data model for the generic CWMP-managed device SHALL apply to the IG, AG and WAN-Gateway FEs.

- TR-104 **[TR104]** that defines the data model for the "SIP end-point" SHALL apply to the IG (see RMS2 functional block of the "Open IPTV Forum – Functional Architecture" document **[ARCH]**).

5.3.5.1.2 UNI-RMS for OITF

Although the remote management functions are specified in the general framework document TR-069 **[TR069]** by the Broadband Forum, the protocol to remotely manage OITF retail devices is intended to support limited functions mainly for Performance Monitoring and Diagnostics. Consequently, an OITF device doesn't fulfil all the requirements that are requested in TR-069 **[TR069]**. The limitations outlined in the following sections shall apply.

OITF RPC Methods Support Requirements

An OITF SHALL implement the following RPC methods:

Method name	OITF requirement	ACS requirement
CPE methods	Responding	Calling
GetRPCMethods	REQUIRED	REQUIRED
SetParameterValues	REQUIRED	REQUIRED
GetParameterValues	REQUIRED	REQUIRED
SetParameterAttributes	REQUIRED	OPTIONAL
GetParameterAttributes	REQUIRED	OPTIONAL
ACS methods	Calling	Responding
Inform	REQUIRED	REQUIRED

As an OITF device doesn't support all the RPC requirements as defined in **[TR069]**, the ACS SHALL implement the GetRPCMethods to discover the limited set of methods supported by the OITF.

The OITF RPC Methods SHALL respect the calling arguments and type as defined in **[TR069]**, with the following definition of the DeviceIdStruct that is used for the DeviceId argument of the Inform method:

- the 3 parameters ManufacturerOUI, ProductClass and Serial Number have slightly different semantic meanings in the context of OIPF and are obtained from the deviceID identifier (refer to section 6.3.2.1, "User Identity Modelling")
    - ManufacturerOUI = HEX(first 3 bytes of SHA-1(X))
    - ProductClass = "OIPF"
    - SerialNumber = HEX(remaining bytes, from 4th on, of SHA-1(X))

  where, X = (MAC address as bytes) + (domain name in ASCII characters).

Name	Type	Description
Manufacturer	String(64)	Manufacturer of the device
OUI	String(6)	In the context of OIPF, this parameter is the hexadecimal value of the first 3 bytes of SHA-1(X)

ProductClass	String(64)	In the context of OIPF, this parameter is always "OIPF"
SerialNumber	String(64)	In the context of OIPF, this parameter is the hexadecimal value of the remaining bytes (from 4th on) of SHA-1(X)

OITF Data Model

In the framework of the Open IPTV Forum, a specific data model for the Remote Management of a retail OITF device has been defined. The data model has been obtained from TR-135 **[TR135]** and TR-106 **[TR106]** with a selection of a reduced set of parameters using the same semantics (with a few exceptions) and the same types. The OITF data model is fully described in 0, “OITF-specific TR-135 and TR-106 Remote Management Objects”.

5.3.5.1.3 Configuration of the IG via Configuration File

CPE WAN Management protocol based on Broadband Forum TR-069 **[TR069]** SHALL be used to configure the IPTV application in the IG. An IPTV configuration file SHALL be used to populate the IG with the list of users with their IMPU, Alias and Passwords and also configure whether user authentication is to be performed by the IG. If GBA Authentication is supported by the IG, the IG SHALL be configured whether it has to provide an intended identity or not in the GBA authentication procedure as described in section 5.3.6.2.2. The file is downloaded to the IG during the IG power up procedure.

The configuration data SHALL be defined in XML and shall include the XML schema to be enforced against the configuration data.

5.3.5.1.3.1 Call Flow

There are 2 cases to be considered; the first case is when the remote server requests the IG to download the configuration file at power up of the IG. This requires the IG to contact the remote server. The download request is subsequently used by the server to request the IG to download the configuration file. Alternatively, if the server is configured (by some means) with the address of the IG, it can request the IG to contact it using the Connection Request Notification mechanism, if the remote server supports this mechanism.

The second case is when the process is initiated by the IG if it detects a corrupted file or if for some reason it lost the file due to a reboot or an internal error.

Figure 3 is a call flow depicting the configuration procedure.

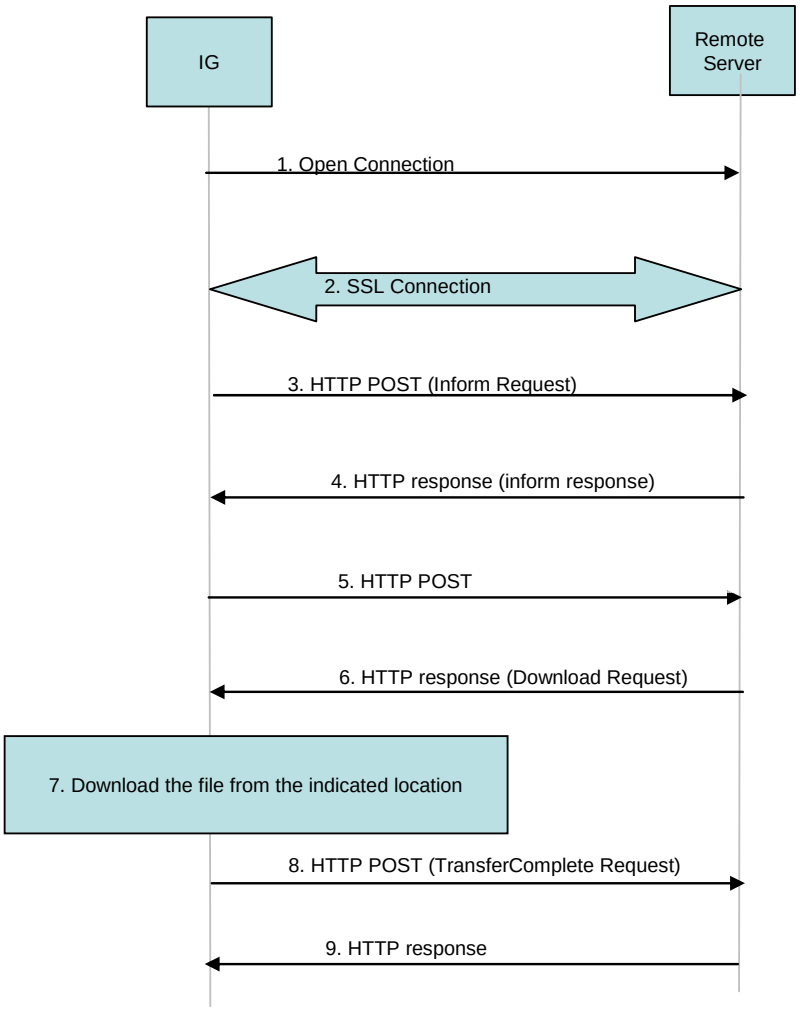


Figure 3: Sequence for the Configuration of an IG

The following is a brief description of the flow:

Steps 1-4: Normal steps as per TR-069.

Step 5: The IG sends an HTTP POST request with no HTTP entity body to the remote server.

Step 6: The server returns an HTTP response that includes a Download request in the HTTP entity body. The arguments are set as follows:

CommandKey:	Mandatory – set by remote server.
FileType:	Mandatory – set to 3: Vendor Configuration File. The vendor in this case is Open IPTV Forum
URI:	Mandatory– set by remote server
Username:	Optional – If used, must be configured in the IG and remote server
Password:	Optional – If used, must be configured in the IG and remote server
TargetFileName:	Mandatory – IPTV-ConfigurationParameters
DelaySeconds:	Mandatory – set to no delay
Successful URI:	Not provided
Failure URI:	Not provided

Step 7: Following that, the IG proceeds to download the configuration file.

Step 8-9: Once the download is complete, the IG sends a TransferComplete request to the remote server. The arguments in the request are set as follows:

CommandKey:	Mandatory - Set to the value received in the Download request.
FaultStruct:	Mandatory in case of failure according to TR-069
StartTime:	Mandatory – Set according to TR-069
FinishTime:	Mandatory – Set according to TR-069

Note that the above sequence is an example and there are other valid sequences that can achieve the same result.

5.3.5.1.3.2 Syntax of the IPTV-Configuration file

The following XML document is an example of a schema for an IPTV-Configuration file

Note that other configurations files with other schemas may also apply to the IG and this is only an example.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="urn:oipf:config:ig:2009"
  xmlns:tns="urn:oipf:config:ig:2009"
  xmlns:enum="urn:ietf:params:xml:ns:enum-token-1.0"
  xmlns="http://www.w3.org/2001/XMLSchema">

  <!—schema filename is config-ig.xsd -->
  <annotation>
    <documentation xml:lang="en">
      This schema is copyrighted by the Open IPTV Forum ("OIPF") and distributed in conjunction
      with Release 1 of the IPTV Solution Specification.

      Disclaimer
      The Open IPTV Forum members accept no liability whatsoever for any use of this document.
      This specification provides multiple options for some features. The Open IPTV Forum Profiling
      specification will complement the Release 1 specifications by defining the Open IPTV Forum
      implementation and deployment profiles. Any implementation based on Open IPTV Forum
      specifications that does not follow the Profiling specifications cannot claim Open IPTV Forum
      compliance.

      Copyright Notification
      No part may be reproduced except as authorized by written permission.
      Any form of reproduction and/or distribution of these works is prohibited.
      Copyright 2009 © Members of the Open IPTV Forum
      All rights reserved.
    </documentation>
  </annotation>

  <import namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="xml.xsd" />
  <import namespace="urn:ietf:params:xml:ns:enum-token-1.0"
   schemaLocation="imports/enum-token-1.0.xsd />

  <element name="IGconfiguration" type="tns:IGconfigurationType" />
  <complexType name="IGconfigurationType">
    <sequence>
      <element name="AuthenticationSet"
        type="tns:AuthenticationSetType" maxOccurs="unbounded" />
      <element name="GatewayAuthentication" type="boolean"
        minOccurs="0" />
      <any namespace="##other" processContents="skip"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
  </complexType>

  <complexType name="AuthenticationSetType">
```



```
    <sequence>
      <element name="Identifier" type="tns:IMSPublicIdType" />
      <element name="Password" type="string" />
      <element name="Alias" type="string" />
      <sequence minOccurs="0">
        <element name="IMPI" type="string" />
        <element name="SIPDigestPassword" type="string" />
      </sequence>
    </sequence>
  </complexType>

  <!-- ================== Definition for IMSPublicIdType ====================-->
  <complexType name="IMSPublicIdType">
    <choice>
      <element name="e164Number" type="enum:e164NumberType" />
      <element name="SIPURI" type="tns:SIPURIType" />
    </choice>
  </complexType>

  <simpleType name="SIPURIType">
    <annotation>
      <documentation xml:lang="en">
        SIP URI pattern is defined based on the SIP URI
        description provided in RFC 3261 (Section 2)
      </documentation>
    </annotation>
    <restriction base="string">
      <pattern
        value="[sS][iI][pP][sS]?:(//([^/?#]*))?([^?#]*)(\?([^#]*))?(#(.*))?" />
    <restriction>
  </simpleType>
</schema>
```

The schema establishes a binding between an IMS Public Identity (IMPU), a user alias and a password. If SIP Digest authentication is used for user authentication, the IMPI and SIPDigestPassword SHALL be included. Otherwise, it si assumed that user authentication is based upon IMS AKA.

The schema also supports a mechanism to instruct the IG if user authentication is mandatory in the Consumer Network.

The schema is extensible.

An example of a configuration file that conforms to the above schema is as follows:

```
<IGconfiguration>
  <AuthenticationSet>
    <Identifier>sip://operator.example.com/MickJ</Identifier>
    <Password>RollingStones</Password>
    <Alias>Mick Jagger</Alias>
    <IMPI>household123@operator.com</IMPI>
    <SIPDigestPassword>CCXDFGGH</SIPDigestPassword>
  </AuthenticationSet>
  <AuthenticationSet>
    <Identifier>sip://operator.example.com/BruceS</Identifier>
    <Password>TheBoss</Password>
    <Alias>BruceSpringstein</Alias>
    <IMPI>household123@operator.com</IMPI>
    <SIPDigestPassword>CCXDFGGH</SIPDigestPassword>
  </AuthenticationSet>
  <GatewayAuthentication>Yes</GatewayAuthentication>
</IGconfiguration>
```

5.3.5.2 Remote Management using DAE APIs

See DAE Specification **[DAE]** section 7.11.5.

5.3.6 User Registration and Network Authentication

5.3.6.1 Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface

5.3.6.1.1 User Registration

This procedure SHALL be invoked in following cases:

- When the OITF is turned on if the OITF supports the native HNI-IGI function.
- When an IPTV end user explicitly logs on at an OITF using an Alias or IMPU other than the default IMPU or Alias.

The IG SHALL NOT perform IMS registration when the IMPU is already registered; however, the IG SHALL maintain a binding between the Alias/IMPU and the new contact address (OITF IP address).

If the identity being registered is not the default identity and if the default identity is not bound to any OITF in the consumer network, then the IG SHALL deregister the default identity at the end of this procedure.

Step 1: The OITF SHALL send an HTTP POST request to the IG on the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <List of HTTP headers> - as per RFC 2616 [HTTP]
  - <list of SIP headers encoded as HTTP headers> - as per Table 25
- **HTTP Request Body:** Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers needed for the outgoing registration message, as per Table 25. The IG shall reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for the rejection.

Step 3: Once the IG completes the IMS registration process, the IG SHALL return a HTTP 200 OK response (or other appropriate responses) to the OITF. The response SHALL includes a list of SIP headers as per Table 26 in additional to the normal HTP headers as per RFC 2616 [HTTP].

If the OITF does not support native HNI-IGI, user registration SHALL be done through a DAE application.

5.3.6.1.2 User De-registration

This procedure is invoked in the following cases:

- The OITF is turned off and the OITF supports native HNI-IGI.
- An IPTV end user, who has registered with his own IMPU, deregisters from an OITF

Note that if the de-registered identity is the default identity for the subscription, and if there are other OITFs in the consumer network that are still turned on, the IG SHALL NOT perform the IMS de-registration procedure. Rather, the IG SHALL remove the binding between the subscription default identity and the OITF's contact address.

The IG SHALL NOT perform IMS deregistration when an IMPU is already registered on multiple OITFs, but the IG SHALL remove the binding between the IMPU and the OITF IP address from which the user has deregistered.

The IG SHALL perform the IMS deregistration procedure if the IMPU was bound to a single OITF.

Step 1: The OITF SHALL send to the IG an HTTP POST request containing an X-OITF-Request-Line header on the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header** including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**

- <list of SIP headers encoded as HTTP headers> - as per Table 25

- **HTTP Request Body:** Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers needed for the outgoing de-registration message as per Table 25. The IG SHALL reject any request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for the rejection.

Step 3: Once the IG completes the IMS de-registration process, the IG SHALL return a HTTP 200 OK response (or other appropriate responses) to the OITF. The response SHALL include a list of SIP headers as per Table 26 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Table 25: List of mandatory HTTP extension headers for User Registration/De-Registration (OITF→IG)

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Request-Line The Request-URI is that of the P-CSCF, and is fetched by the OITF as per section 7.1.1 of ETSI TS 183 019 Network Attachment: User-Network protocol Interface Definitions [TS183019]. The IG SHALL be responsible for resolving the domain name.	RFC 3261 [SIP] REGISTER <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Contact Notes: 1. Contact MUST include Feature Tags parameter. 2. URI parameter MUST be included. 3. Expires parameter SHOULD be included 4. Priority parameter SHOULD be included IG adds all the other mandatory parameters that are absent in the X-OITF-Contact. Default values are assigned by the IG to optional parameters that are not provided in the X-OITF-Contact.	RFC 3261 [SIP] and RFC 3840 [RFC3840]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

Table 26: List of HTTP extension headers for User Registration/De-Registration Response (IG→OITF)

X-OITF SIP Header	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	SIP header field prefixed with X-OITF
X-OITF-To	SIP header field prefixed with X-OITF
X-OITF-Expires	SIP header field prefixed with X-OITF
X-OITF-Contact	SIP header field prefixed with X-OITF
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

If the OITF does nto support native HNI-IGI, user deregistration SHALL be done through a DAE application.

5.3.6.1.3 Procedure for Refreshing a Registration

This procedure MAY be initiated by the OITF at any time before the expiry of the registration refresh timer.

The procedure is the same as the procedure for registering a user. A registration SHALL be terminated if it is not refreshed before the expiry of the registration refresh timer.

For an OITF-initiated registration, the IG SHALL consider a registration terminated (that is, the user de-registered) if it is not refreshed. In this case, the IG executed the procedures associated with user deregistration.

5.3.6.1.4 Procedure for Subscription to the Registration Event Package

This procedure SHALL be invoked immediately after the successful registration of an IMPU (including the default identity) or an IPTV end-user identity.

Step 1: The OITF SHALL send an HTTP POST request to the IG on the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 27
- **HTTP Request Body:** Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers for the outgoing subscription request message, as per Table 27. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for the rejection.

Step 3: The IG SHALL send a SIP SUBSCRIBE to the network, to subscribe to the Registration event, and shall wait for the response to the subscription request. The IG SHALL return a HTTP 200 OK response (or other appropriate responses) to the OITF to report the response to the subscription request. The response SHALL include a list of SIP headers as per Table 28 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Step 4: Following that, the OITF SHALL send an HTTP HNI-IGI PENDING_IG request (refer to section 5.5.1.1, "HNI-IGI Message Types"), and shall wait for any response.

Step 5: When a SIP NOTIFY is received by the IG, the IG SHALL return a HTTP 200 OK response to the OITF. The response SHALL include the list of SIP headers as per Table 29 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**. The body of the HTTP response SHALL include the SIP body received in the incoming NOTIFY (See also section 6.3.2.2, "Procedure for User Registration and Authentication in a Managed Model on UNIS-8.)"

Step 6: Once the OITF accepts the incoming SIP NOTIFY, it SHALL send an HTTP POST PENDING_IG request to the IG. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 30
- **HTTP Request Body:** Empty

Step 7: The IG SHALL send the SIP 200 OK response to the network and then SHALL return to Step 5 to handle any subsequent NOTIFY received from the network.

Table 27: Supported HTTP extension headers in the HNI-IGI SUBSCRIBE Request for the Registration Event Package

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Request-Line Note: The request URI SHALL be set to the Public identity of the IPTV end user who has just registered	RFC 3261 [SIP] SUBSCRIBE <Request URI> SIP/2.0)
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]

X-OITF-Event	RFC 3265 [SIP]and RFC 3680 (registration event) [SIP-REG]
X-OITF-Accept	RFC 3265 [SIP-EVNT] and RFC 3680 **[SIP-REG]**
X-OITF-Contact Notes: 1. URI parameter SHALL be included, and SHALL match what is sent in the Contact header included in the registration request. 2. Expires parameter SHOULD be included 3. Priority parameter SHOULD be included The IG includes all other mandatory parameters that are absent.	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
XOITF-Expires	RFC 3261 [SIP]

Table 28: Supported HTTP extension headers in the response to an HNI-IGI SUBSCRIBE Request for the Registration Event Package

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Expires	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]

Table 29: List of HTTP extension headers for a HNI-IGI NOTIFY request sent IG→OITF

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Notes: The Request URI MUST match the contact URI included in the contact field of the SIP SUBSCRIBE	RFC 3261 [SIP] NOTIFY <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Event	RFC 3265 [SIP-EVNT] and RFC 3680 (registration event) [SIP-REG]
X-OITF-Call-ID	RFC 3265 [SIP-EVNT] and RFC 3680 [SIP-REG]
X-OITF-Subscription-State	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3265 [SIP-EVNT] and RFC 3680 [SIP-REG]
X-OITF-Content-Length	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]

Table 30: List of HTTP extension headers in the response to a NOTIFY request

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]

X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]

5.3.6.1.5 Procedure for Terminating a Subscription to the Registration Event Package

This procedure SHALL be invoked prior to de-registering a user

The procedure is the same as the procedure for initiating a subscription to the Registration event, however in this case the X-OITF-Expires header in Table 27 SHALL be set to 0.

For an OITF-initiated registration, the IG SHALL consider a subscription terminated if is not refreshed.

5.3.6.1.6 Refreshing Subscription to Registration Event

The procedure is the same as the procedure for initiating a subscription.

It is the responsibility of the application initiating the subscription procedure to refresh the subscription according to the refresh subscription timer information received in the response to the subscription request. Refreshing the subscription SHOULD be performed before the expiry of the refresh timer. A subscription that is not refreshed before the expiration of the refresh timer SHALL be terminated

5.3.6.1.7 Registration of DAE/Embedded Applications

IMS applications, DAE or embedded, that are initiated in the OITF and expect unsolicited incoming messages SHALL register with the IMS network the feature tags and/or the appropriate service URN (ICSI) and /or IMS application reference identifier (IARI) for the initiated application where mandated by the specification governing the application **[TS183063]**, **[SMPL-IM]**, **[TS124503]**, **[RFC3840]**, **[RFC3841]**. This allows unsolicited incoming SIP messages destined for users and targeted for these applications to be delivered to the appropriate application instance in the OITF.

The procedure used by an application for registering the appropriate feature tags and/or service URN (ICSI) and/or IARI is the same procedure used for user registration.

5.3.6.2 GBA Authentication

This section describes the HNI-IGI message for the GBA Authentication. For the details of the sequence for GBA Authentication, refer to section 5.4.4 of **[CSP]**. Note that GBA authentication applies only for user registration and authentication based on IMS AKA.

5.3.6.2.1 Initial GBA registration

After IMS registration is successfully performed, and if the IG supports GBA Authentication, OITFs supporting native HNI-IGI SHALL issue following GBA registration request to the IG. OITFs that do not support native HNI-IGI do not support GBA.

Step 1: The OITF SHALL send an HTTP POST request to the IG. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Headers:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-HNI-IGI-Request: GBA-Registration>
- **HTTP Request Body:** Empty

Step 2: After the GBA bootstrapping procedure over UNIS-9, the IG returns an HTTP 200 OK response.

5.3.6.2.2 Credential Retrieval by an OITF for Re-use of GBA Authentication

The key Ks that is established during the GBA registration MAY be reused later for user authentication and service access by consumer network applications.

Each time an OITF needs to access a service that is offered by an AS (i.e. NAF) that requires GBA Authentication, a specific key Ks_NAF SHALL be derived by the IG and the server side GBA Single Sign-on function (the BSF). This generated key SHALL be conveyed to the OITF in the consumer network by the IG, and to the AS by the server side GBA Single Sign-on function (the BSF). The key Ks_NAF SHALL then be used for authentication between the OITF and the AS, using HTTP Digest authentication as specified by **[UB-UA]**. The OITF SHALL act as the UE as specified in **[UB-UA]**.

As a pre-requisite to this procedure, the GBA procedure MUST have been successfully completed.

The complete procedure for retrieval of credentials by the OITF from the IG is specified in **[CSP]**.

The HNI-IGI procedure for credential retrieval is as follows:-

Step 1: The OITF SHALL send an HTTP POST request to the IG. The request includes the NAF_ID. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Headers:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-HNI-IGI-Request> - set to Fetch-GBA-Credentials
  - <X-HNI-IGI-NAF-FQDN> - set to NAF FQDN extracted from the HTTP authentication realm as specified in [UB-UA].
- **HTTP Request Body:** Empty

Step 2: The IG SHALL generate Ks_NAF, which is computed as follows:

Ks_NAF = KDF (Ks, “gba-me”, RAND, IMPI, NAF_ID), where KDF is the key derivation function as specified in Annex B of **[GAA]** and the key derivation parameters consist of the user's IMPI, the NAF_ID and RAND. The NAF_ID is constructed as follows: NAF_ID = FQDN of the NAF || Ua security protocol identifier as specified in 3GPP 33 220 [GAA]. The identifier for Ua security protocol HTTP Digest authentication according to 3GPP 33 220 [GAA] is (0x01, 0x00, 0x00, 0x00, 0x02).

The IG SHALL return an HTTP 200 OK to the OITF that includes the Ks_NAF, the B-TID, the lifetime of the key Ks_NAF, and optionally the intended identity. The lifetime indicates the expiry time of the key Ks_NAF and is equal to the lifetime of the key Ks (which was specified by the BSF during the GBA bootstrapping procedure). The content of the HTTP 200 OK response is as follows:

- **HTTP Response Headers:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-HNI-IGI-KS_NAF> - set to the computed Ks_NAF
  - < X-HNI-IGI-B_TID> - set to the B-TID
  - <X-HNI-IGI-LifeTime> - set to life time of the key Ks_NAF
  - <X-HNI-IGI-Intended-Identity> - set to the intended identity. This header is optional and its use is described in [CSP].

5.3.6.3 User ID Retrieval for managed network services

The OITF SHALL retrieve a list of user IDs (IMPU and Alias) from the IG for managed service over the HNI-IGI interface. This procedure SHOULD NOT require user authentication. The IG SHALL at a minimum provide the default identity for the household and MAY provide all available identities. OITFs that do not support native HNI-IGI SHOULD retrieve the User IDs using DAE.

Step 1: The OITF SHALL send an HTTP POST request to the IG. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-HNI-IGI-Request: Fetch-UserIDs>
- **HTTP Request Body:** Empty

Step 2: The IG returns a list of user IDs (IMPU and Aliases) as follows:

The IG SHALL return an HTTP 200 OK to the OITF. The content of the HTTP 200 OK response SHALL be as follows:

- **HTTP Response Headers:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
- **HTTP Response Body:** a list of IMPUs and Display names (alias). Elements are separated with commas, entries are separates with semi-colon, in the format <IMPU1>,<alias1>;<IMPU2>,<alias2>,… etc. The first entry SHALL be the default identity for the household.

  The usage of IMPU and Alias by the OITF is defined by the CSP specification.

  Depending on the policy of the IG and service provider, the IG MAY return the default identity only. In this case, the user of the OITF SHALL be required to enter a user ID manually.

5.4 Protocols for Communications Functions

5.4.1 CallerID

5.4.1.1 Procedure for Instant Message Based Caller ID

5.4.1.1.1 Procedure on HNI-IGI

The OITF supports the following procedure for Caller ID. The incoming message carrying a Caller ID can either be handled by a native application in the OITF, or in a DAE application. The same HNI-IGI message format is used in either case.

Step 1: The IG receives an incoming SIP MESSAGE from the network.

Step 2: The IG forwards the information in the SIP MESSAGE to the OITF in the HTTP 200 OK response to a PENDING_IG request that was established when the application started. The list of SIP headers to be included in the message to the OITF SHALL be as per Table 31. The body of the SIP MESSAGE SHALL be included in the HTTP response body.

Step 3: Upon receipt of the message, the OITF SHALL issue an HTTP POST request. The content of the HTTP request SHALL be as follows:

- **HTTP Request Header:** including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 32
- **HTTP Request Body:** Empty

Step 4: The IG SHALL send SIP 200 OK to the network.

Note: For handling of new incoming SIP MESSAGE, refer to section 5.3.2 of the DAE specification **[DAE]** titled “IMS Notification Framework”

Table 31: List of HTTP extension headers for an Instant Message Based Caller ID (IG→OITF)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Note: The request URI MUST be set to the IMS Public Identity (IMPU) of the target of the message	RFC 3261 [SIP] MESSAGE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3428 **[SIP-IM]**, Draft OMA-TS-SIMPLE_IM-V1_0-20080820-D **[SMPL-IM]**
X-OITF-Content-Length	RFC 3261 [SIP]

Table 32: List of HTTP extension headers for the response to an Instant Message Based Caller ID (OITF→IG)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

5.4.1.2 Procedure for IMS Telephony Based Caller ID (OPTIONAL)

5.4.1.2.1 Procedure for HNI-IGI

The following procedure MAY be supported in the OITF for Caller ID presentation to the OITF user as a result for an incoming IMS voice call to the IG.

The incoming message, carrying information on the IMS voice call, can either be handled by a native application in the OITF, or by a DAE application. The same HNI-IGI message format is used in either case.

Step 1: The IG receives an incoming SIP INVITE.

Step 2: The IG forwards the SIP INVITE to the OITF as an HTTP response to a PENDING_IG request. The list of SIP headers to be included in the message to the OITF shall be as per Table 33. The content of the invite message SHALL also be included.

Step 3: Upon receipt of the message, the OITF issues an HTTP POST request indicating that the voice call is not supported by the OITF by response code 415 Unsupported Media Type. Other values MAY be used according to RFC 3261 **[SIP]**. The content of the HTTP Request is as follows:

- **HTTP Request Header** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 34
- **HTTP Request Body:** application/sdp

Step 4: The IG SHALL forward the SIP response to the network.

Step 5: When the IG receives the SIP ACK from the network and SHALL forward it to the OITF as an HTTP response to a PENDING_IG request. The list of SIP headers to be included in the message to the OITF shall be as per Table 35.

Note: For handling of new incoming INVITE messages for new dialogs, refer to section 5.3.2 of the DAE specification entitled "IMS Notification Framework"

Table 33: List of HTTP extension headers on the HNI-IGI interface (IG→OITF) for a received SIP INVITE

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Note: The request URI MUST be set to the IMS Public User Identity of the target of the message	RFC 3261 [SIP] INVITE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP], ETSI ES 283 002 [TS124503]
X-OITF-P-Called-Party-ID	ETSI ES 283 002 [TS124503]
X-OITF-P-Asserted-Identity	ETSI ES 283 002 [TS124503]

Table 34: List of HTTP extension headers on the HNI-IGI interface (OITF→IG) for a response to the SIP INVITE

X-OITF HTTP Header	Source of Coding Information
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Accept	RFC 3261 [SIP]

Table 35: List of HTTP headers in the HNI-IGI ACK Message (IG→OITF)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line	RFC 3261 [SIP] ACK <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Accept	RFC 3261 [SIP]

5.4.2 Instant Messaging

5.4.2.1 Procedure for Instant Messaging on HGI INI

Instant Messaging on the OITF uses the HNI-IGI functionality, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)."

There are two cases, messages originating from the OITF, and messages terminating in the OITF.

5.4.2.1.1 Procedure for OITF Originating an Instant Messaging

The following procedure is supported in the OITF to originate instant messages:

An instant message can either originate from a native application in the OITF or from a DAE application. The same HNI-IGI message format is used.

Step 1: The OITF SHALL send an HTTP POST request to the IG using the HNI-IGI functionality, as described in 5.5.1, OITF-IG interface (HNI-IGI). The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 36
- **HTTP Request Body:** The content type as per RFC 3428 [SIP-IM]

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers for the message as per Table 36. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL send a SIP MESSAGE to the network. When the IG receives the response, the IG SHALL return a 200 OK HTTP response (or other appropriate responses) to the OITF to report the response to the SIP MESSAGE. The response includes a list of SIP headers as per Table 37 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Table 36: List of HTTP extension headers for an outgoing Instant Message (OITF→IG)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Note: The request URI MUST be set to the Public identity of the target of the message	RFC 3261 [SIP] MESSAGE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]

Table 37: List of HTTP extension headers for the response to an outgoing and incoming Instant Message (IG→OITF and OITF→IG)

SIP Headers	Source of Coding Information
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]

5.4.2.1.2 Incoming Instant Messaging Procedure

The following procedure is supported in the OITF for incoming instant messages:

The incoming message can be handled either by a native application in the OITF, or in a DAE application. The same HNI-IGI message format is used in either case.

Step 1: The IG receives an incoming SIP MESSAGE

Step 2: The IG SHALL forward the SIP MESSAGE to the OITF as an HTTP response to a PENDING_IG request. The list of SIP headers to be included in the notification forwarded to the OITF SHALL be as per Table 38. The body of the SIP MESSAGE SHALL be included in the HTTP body.

Step 3: Upon receipt of the message, the OITF SHALL issue an HTTP POST request. The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 37
- **HTTP Request Body:** Empty

Step 4: The IG SHALL forward the SIP 200 OK to the network.

Table 38: List of HTTP extension headers for an Incoming Instant Message (IG→OITF)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Note: The request URI MUST be set to the Public identity of the target of the message	RFC 3261 [SIP] MESSAGE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP], Draft OMA-TS-SIMPLE_IM-V1_0-20080820-D **[SMPL-IM]**
X-OITF-Content-Length	RFC 3261 [SIP]

5.4.3 IM Session (Chat using MSRP)

5.4.3.1 Procedure for initiating an Instant Messaging Session (MSRP Chat)

To initiate a chatting session using MSRP, the OITF SHALL use the following procedure:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 39
- **HTTP Request Body:** Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers as per Table 39. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection. The IG SHALL generate the INVITE by mapping the X-OITF headers to

the appropriate SIP header. As the IG implements MSRP, the IG SHALL include all the necessary additional SIP headers and the SDP body to initiate the MSRP session as follows:

- The Content-Type header SHALL be added and set to "application/sdp"
- The Content-Length header SHALL be added and set to the appropriate value
- The message body SHALL include the following information:
  - A, c = IN IP4 <IP address> , where <IP address> would contain the IP address of the IG,
  - An, m = message <tcp port> tcp/msrp, where tcp port is a TCP port could be set to the dummy value "9"
  - An, a = accept-types:message/cpim, attribute which is mapped from the "X-OITF-Accept:" header value
  - An a = path msrp://<IP address>:<tcpport>/<session-id>; tcp, where:
    - <IP address> would contain the IP address of the IG
    - <tcpport> would be assigned automatically by the IG
    - <session-id> would be assigned automatically by the IG and bound to the requesting OITF Chatting application

NOTE: In this case the IG is not service agnostic. The IG detects that this session is for MSRP by examining the X-OITF-Accept header which SHALL include message/cpim (See example in section C.2.1.2, "Chat.")

Step 3: The IG SHALL send a HTTP 200 OK response to the OITF when the SIP 200 OK is received as a response to the session invitation. The SIP 200 OK headers are mapped as indicated in Table 40, in addition to the normal HTTP 200 OK headers. The IG SHALL not forward the body of the SIP 200 OK to the OITF. The IG SHALL establish and maintain the MSRP state information including the binding between the logical entities (indicated in the From and To headers) and the corresponding path (the one initiated by the IG for the OITF and the one indicated by the distant entity for the To:). The IG SHALL maintain a binding between the SIP dialog and the MRSP state information for the duration of the SIP dialog.

Step 4: Upon receipt of a 200 OK response, the OITF SHALL send an HTTP PENDING_IG to acknowledge the final response.

The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 41
- **HTTP Request Body:** Empty

Table 39: List of HTTP extension headers for IM INVITE request (OITF→IG)

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Request-Line The request URI SHALL be set to the IMPU of the subscriber with whom the session is requested.	RFC 3261 [SIP] INVITE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To MUST be set to the value of the request URI in the "X-OITF-Request-Line INVITE" header	RFC 3261 [SIP]

X-OITF-Contact Notes: URI parameter SHALL be included and SHALL match what is sent in the Contact header included in the registration request. Expires parameter SHOULD be included	RFC 3261 [SIP]
X-OITF-Accept-Contact	Set according to [SMPL-IM]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Accept SHALL be set to: "message/cpim"	[SMPL-IM]

Table 40: List of HTTP extension headers for a 200 OK response received for the INVITE IG→OITF

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]
X-OITF-Accept	RFC 3261 [SIP]

Table 41: List of HTTP extension headers in HNI-IGI ACK Request

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Request-Line The Request-URI in the ACK request shall be the contact included in the response to the INVITE message	RFC 3261 [SIP] ACK <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Contact The URI parameter MUST be included, and MUST match what has been inserted in the INVITE message. IG includes all other mandatory parameters that are absent.	RFC 3261 [SIP]

5.4.3.2 MSRP Invocation

The OITF shall access MSRP capabilities in the IG using the X-HNI-IGI headers.

5.4.3.2.1 Outgoing MSRP Chat Messages

The OITF SHALL send an outgoing MSRP chat message using the following procedure:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of HNI-IGI headers encoded as HTTP headers> - as per Table 42

- **HTTP Request Body:** The Message in plain text.

Step 2: The IG SHALL validate that the request includes all the mandatory HNI-IGI headers for the process as per Table 42. The IG SHALL reject a request that is missing any mandatory HNI-IGI headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL validate the Call-ID and the Message-ID, if present, and subsequently SHALL send an MSRP SEND message to the network, then wait for the MSRP 200 OK response from the network. The IG SHALL return a 200 OK HTTP response to the OITF when it receives the MSRP 200 OK (or other responses). The 200 OK HTTP response SHALL include the HNI-IGI headers as per Table 43 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Table 42: List of HNI-IGI HTTP extension headers for an MSRP SEND Request (OITF→IG)

X-HNI-IGI HTTP Header	Source of Information for Coding purposes
X-HNI-IGI-Request SEND MESSAGE	[SMPL-IM]
X-HNI-IGI-Message-ID	SHALL be left blank for the first message.
X-HNI-IGI-Call-ID	SHALL be set to the same value for the INVITE transaction that initiated the session
X-HNI-IGI-From	SHALL be set to the identity of the originator of the message
X-HNI-IGI-To	SHALL be set to the identity of the recipient of the message

Table 43: List of HNI-IGI HTTP extension headers included in the HTTP 200 OK response (IG→OITF)

X-HNI-IGI Headers	Source of Information for Coding purposes
X-HNI-IGI-Response MSRP <Response>	[SMPL-IM]
X-HNI-IGI-Message-ID	[SMPL-IM]
X-HNI-IGI-From	[SMPL-IM]
X-HNI-IGI-To	[SMPL-IM]

5.4.3.2.2 Sending an MSRP Chat State Message

The OITF SHALL use the following procedure to indicate the activity of the user (e.g., “Is Composing”):

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, “OITF-IG Interface (HNI-IGI).” The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of HNI- IGI headers encoded as HTTP headers> - as per Table 44
- **HTTP Request Body:** SHALL contain the appropriate XML document as indicated in RFC 3994 [RFC3994] and OMA-TS-SIMPLE-IM_V1_0-20080820-D [SMPL-IM].

Step 2: The IG SHALL validate that the request includes all the mandatory HNI-IGI headers for the process as per Table 44. The IG SHALL reject a request that is missing any mandatory HNI-IGI headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL validate the Call-ID and the Message-ID, and send an MSRP SEND message to the network after performing the necessary mapping and adding the appropriate tags. The IG SHALL then wait for the MSRP 200 OK response from the network. The IG SHALL return a 200 OK HTTP response (or

other appropriate responses) to the OITF when it receives the MSRP 200 OK (or other responses). The response SHALL include a list of HNI-IGI headers as per Table 43 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Table 44: List of HNI-IGI HTTP extension headers for an MSRP SEND ACTIVITY Request (OITF→IG)

X-HNI-IGI HTTP Header	Source of Information for Coding purposes
X-HNI-IGI-Request MSRP SEND ACTIVITY	OMA-TS-SIMPLE_IM-V1_0-20080820-D [SMPL-IM]
X-HNI-IGI-Message-ID	SHALL be set to the appropriate message id
X-HNI-IGI-Call-ID	SHALL be set to the same value for the INVITE transaction that initiated the session
X-HNI-IGI-From	**[SMPL-IM]**
X-HNI-IGI-To	**[SMPL-IM]**

5.4.3.2.3 Receiving an MSRP Chat Message

The IG SHALL use the following procedure when receiving an incoming MSRP message:

Step 1: In response to a PENDING_IG request, the IG SHALL send an HTTP 200 OK response to the OITF over the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The response SHALL include the HNI-IGI headers listed in Table 45, in addition to the mandatory HTTP headers in RFC 2616 **[HTTP]**. The body of the HTTP 200 OK response SHALL include the received text.

Step 2: The OITF SHALL respond with an HTTP POST request with its body containing an MSRP 200 OK response. The contents of the HTTP Request shall be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of HNI-IGI headers encoded in HTTP headers> - as per Table 46
- **HTTP Request Body:** Empty

Step 3: The IG SHALL send the response from the OITF in an MSRP response message to the network after performing the necessary validation.

Table 45: List of HNI-IGI HTTP extension headers for an incoming MSRP message (IG→OITF)

X-HNI-IGI HTTP Header	Source of Information for Coding purposes
X-HNI-IGI-Request MSRP RECEIVE MESSAGE	[SMPL-IM]
X-HNI-IGI-Message-ID	SHALL be set to the appropriate message id
X-HNI-IGI-Call-ID	SHALL be set to the same value for the INVITE transaction that initiated the session
X-HNI-IGI-From	SHALL be set to the remote user
X-HNI-IGI-To	SHALL be set to the recipient of the message

Table 46: List of HNI-IGI HTTP extension headers for an MSRP 200 OK Response to an incoming MSRP Message (OITF→IG)

X-HNI-IGI HTTP Header	Source of Information for Coding purposes
X-HNI-IGI-Response MSRP <Response>	Set to the appropriate Response

X-HNI-IGI-Message-ID	SHALL be set to the appropriate message id

5.4.3.2.4 Receiving an MSRP Chat State Message

The IG SHALL use the following procedure when receiving an incoming MSRP Chat State message:

Step 1: In response to a PENDING_IG request, the IG SHALL send an HTTP 200 OK response to the OITF over the HNI-IGI interface, as described in OITF-IG Interface (HNI-IGI). The response SHALL include the HNI-IGI headers listed in Table 47 in addition to the mandatory HTTP headers in RFC 2616 **[HTTP]**. The body of the HTTP 200 OK response SHALL contain the appropriate XML document as indicated in RFC 3994 **[RFC3994]** and OMA-TS-SIMPLE-IM_V1_0-20080820-D **[SMPL-IM]**

Step 2: The OITF SHALL respond with an HTTP POST request with its body containing an MSRP 200 OK response. The contents of the HTTP Request shall be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of HNI-IGI headers encoded in HTTP headers> - as per Table 46
- **HTTP Request Body:** Empty

Step 3: The IG SHALL send the response from the OITF in an MSRP response message to the network after performing the necessary validation.

Table 47: List of HNI-IGI HTTP extension headers for an incoming MSRP RECEIVE ACTIVITY (IG→OITF)

X-HNI-IGI HTTP Header	Source of Information for Coding purposes
X-HNI-IGI-Request MSRP RECEIVE ACTIVITY	OMA-TS-SIMPLE-IM-V1-0-20080820-D **[SMPL-IM]**
X-HNI-IGI-Message-ID	SHALL be set to the appropriate message id
X-HNI-IGI-Call-ID	SHALL be set to the same value for the INVITE transaction that initiated the session
X-HNI-IGI-From	SHALL be set to the remote user
X-HNI-IGI-To	SHALL be set to the recipient of the message

5.4.3.3 Terminating an IM Session (MSRP Chat)

In order to terminate an MSRP session, the OITF SHALL use the following procedure:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, “OITF-IG Interface (HNI-IGI).” The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header:** Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 48
- **HTTP Request Body:** Empty

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers for the process as per Table 48. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection. The IG shall generate the SIP BYE by mapping the X-OITF headers to the appropriate SIP headers

Step 3: The IG SHALL send a HTTP 200 OK response to the OITF when the SIP 200 OK is received as a response to the Chat session termination request. The SIP 200 OK headers are mapped as indicated in Table 49 in addition to the normal HTTP 200 OK headers.

Table 48: List of HTTP extension headers for an MSRP BYE request (OITF→IG)

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Request-Line	RFC 3261 [SIP] BYE <Request URI> SIP/ 2.0
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Contact SHALL be set to the value received in the contact of a 200 OK for session termination or SIP INVITE for session origination	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Content-Length Must be set to 0	

Table 49: List of HTTP extension headers for a 200 OK response to a BYE (IG→OITF)

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 200 OK
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Contact	RFC 3261 [SIP]
X-OITF-Content-Length: Set to 0	RFC 3261 [SIP]

5.4.3.4 Remote Termination of an IM Session (MSRP Chat)

The IG SHALL use the following procedure when receiving an incoming SIP BYE message for an ongoing IM session (MSRP Chat):

Step 1: The IG receives a SIP BYE message from the network.

Step 2: The IG forwards the information in the SIP BYE to the OITF over the HNI-IGI interface in the HTTP 200 OK response to a PENDING_IG request. The response SHALL include the list of SIP headers listed in Table 50, in addition to the mandatory HTTP headers in RFC 2616 **[HTTP]**.

Step 3: The OITF SHALL respond with an HTTP POST request. The content of the HTTP Request shall be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded in HTTP headers> - as per Table 51
- **HTTP Request Body :** Empty

Step 4: The IG SHALL send SIP 200 OK to the network.

Table 50: List of HTTP extension headers for an Incoming SIP BYE (IG→OITF)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Note: The Request URI MUST match the contact URI included in the contact field of the SIP INVITE (for outgoing session) or a 200 OK (for incoming session)	RFC 3261 **[SIP]** BYE <Request URI> SIP/ 2.0
X-OITF-From	RFC 3261 **[SIP]**
X-OITF-To	RFC 3261 **[SIP]**
X-OITF-Call-ID	RFC 3261 **[SIP]**
X-OITF-CSeq	RFC 3261 **[SIP]**
X-OITF-Content-Length Must be set to 0	RFC 3261 **[SIP]**
X-OITF-Contact	RFC 3261 **[SIP]**

Table 51: List of HTTP extension headers for the response to an SIP BYE (OITF→IG)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Response-Line	RFC 3261 **[SIP]** SIP/2.0 <response>
X-OITF-From	RFC 3261 **[SIP]**
X-OITF-To	RFC 3261 **[SIP]**
X-OITF-Call-ID	RFC 3261 **[SIP]**
X-OITF-CSeq	RFC 3261 **[SIP]**

5.4.3.5 Procedure for Reception of a remotely initiated Instant Messaging Session (MSRP Chat)

The IG SHALL use the following procedure when receiving an incoming SIP INVITE message for a new IM session (MSRP Chat):

Step 1: The IG receives a SIP INVITE message from the network.

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers as per Table 52. This is required since the IG must send all this information to the OITF. The IG SHALL reject any incoming request that is missing any mandatory parameter. Subsequently, the IG SHALL perform the following checks:

- Ensure that the Content-Type header is present and set to “application/sdp”
- Verify that the SDP body includes the following information:
  - A, c = IN IP4 <IP address>, where <IP address> would contain the remote IP address.
  - An, m = message <tcp port> tcp/msrp, where tcp port is a TCP port and could be set to the dummy value “9”
  - An, a = accept-types:message/cpim, attribute which is mapped from the Accept header value.
  - An a = path msrp://<IP address>:<tcpport>/<session-id>; tcp, where:
    - <IP address> would contain the remote IP address
    - <tcpport> remote IP port
    - <session-id> assigned automatically by the remote peer.

Step 3: Following that, the IG retains and stores information in the SDP, and forwards only the information in the SIP INVITE headers to the OITF over the HNI-IGI interface in the 200 OK HTTP response to a

PENDING_IG request that was sent by the OITF to the IG when the application was launched. The response SHALL include the list of SIP headers listed in Table 52, in addition to the mandatory HTTP headers in RFC 2616 **[HTTP]**.

Step 4: The OITF SHALL respond with an HTTP POST request that includes the OITF response to the incoming INVITE. The content of the HTTP Request shall be as follows:

- **HTTP Request Header**: Including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded in HTTP headers> - as per Table 53
- **HTTP Request Body:** Empty

Step 5: The IG SHALL append the SDP to the SIP 200 OK before sending it to the network. The appended SDP SHALL include the following information:

- A, c = IN IP4 <IP address> , where <IP address> would contain the IP address of the IG,
- An, m = message <tcp port> tcp/msrp, where tcp port is a TCP port could be set to the dummy value "9"
- An, a = accept-types:message/cpim, attribute which is mapped from the "X-OITF-Accept:" header value
- An a = path msrp://<IP address>:<tcpport>/<session-id>; tcp, where:
  - <IP address> would contain the IP address of the IG
  - <tcpport> would be assigned automatically by the IG
  - <session-id> would be assigned automatically by the IG and bound to the responding OITF Chatting application

Step 6: The IG receives a SIP ACK message from the network

Step 7: Following that, the IG SHALL send the information in the incoming ACK message to the OITF in a 200 OK HTTP response. The response includes a list of SIP headers as per Table 54.

Note: Any SDP information is retained in the IG since the IG handles the MSPP protocol

Step 8: The OITF SHALL send an HTTP HNI-IGI PENDING_IG request to the IG and SHALL wait for any incoming messages.

Table 52: List of HTTP extension headers for an incoming IM INVITE request (IG→OITF)

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Request-Line The request URI SHALL be set to the IMPU of the subscriber with whom the session is intended	RFC 3261 **[SIP]** INVITE <Request URI> SIP/2.0
X-OITF-From	RFC 3261 **[SIP]**
X-OITF-To MUST be set to the value of the request URI in the "X-OITF-Request-Line INVITE" header	RFC 3261 **[SIP]**
X-OITF-Contact	RFC 3261 **[SIP]**
X-OITF-Accept-Contact	Set according to **[SMPL-IM]**
X-OITF-Call-ID	RFC 3261 **[SIP]**
X-OITF-CSeq	RFC 3261 **[SIP]**
X-OITF-Accept SHALL be set to: "message/cpim"	**[SMPL-IM]**

Table 53: List of HTTP extension headers for the response to an Incoming IM INVITE Request (OITF→IG)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Response-Line	RFC 3261 **[SIP]** SIP/2.0 <response>
X-OITF-From	RFC 3261 **[SIP]**
X-OITF-To	RFC 3261 **[SIP]**
X-OITF-Call-ID	RFC 3261 **[SIP]**
X-OITF-CSeq	RFC 3261 **[SIP]**
X-OITF-Accept SHALL be set to "message/cpim"	RFC 3261 **[SIP]**
X-OITF-Contact Notes: URI parameter SHALL be included and SHALL match what is returned in the contact header includes in the response to the registration process Expires parameter SHOULD be included	RFC 3261 **[SIP]**

Table 54: Supported HTTP extension headers in HNI-IGI ACK Request for successful IM Session (MSRP Chat) (IG→OITF)

X-OITF HTTP Headers	Source of Information for Coding purposes
X-OITF-Request-Line The Request-URI in the ACK request SHALL be the contact included in the response to the INVITE message	RFC 3261 **[SIP]** ACK <Request URI> SIP/2.0
X-OITF-From	RFC 3261 **[SIP]**
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 **[SIP]**
X-OITF-Call-ID	RFC 3261 **[SIP]**
X-OITF-CSeq	RFC 3261 **[SIP]**
X-OITF-Contact The URI parameter SHALL be included, and SHALL match what been received in the incoming INVITE message.	RFC 3261 **[SIP]**

5.4.4 Presence

5.4.4.1 Procedures for Subscription to Presence on the HNI-IGI interface

The procedure for subscription to the Presence event SHALL be invoked from either a DAE application or an embedded application in the OITF. The procedure is as follows:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header** including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 55
- **HTTP Request Body:** Empty

Step2: The IG SHALL validate that the request includes all the mandatory SIP headers for the subscription process as per Table 55. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL send a SIP SUBSCRIBE to the network, to subscribe to the Presence event, and shall wait for the response to the subscription request. The IG SHALL then return a 200 OK HTTP response to the OITF to report the response to the subscription request. The response includes a list of SIP headers as per Table 56, in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Step 4: The OITF SHALL send an HTTP HNI-IGI PENDING_IG request (refer to section 5.5.1.1, "HNI-IGI Message Types"), and SHALL wait for any incoming messages.

Step 5: When a SIP NOTIFY is received by the IG, the IG SHALL return a 200 OK HTTP response to the OITF containing the information in the incoming NOTIFY message. The response includes a list of SIP headers as per Table 57 in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**. The body of the HTTP response SHALL include the SIP body received in the incoming NOTIFY compliant to Annex E of **[TS183063]**.

Step 6: Once the OITF accepts the incoming SIP NOTIFY, it SHALL send an HTTP POST PENDING_IG request to the IG to acknowledge the receipt of notification and issue a new pending HTTP request. The content of the HTTP request SHALL be as follows:

- **HTTP Request Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 58
- **HTTP Request Body:** Empty

Step 7: The IG SHALL send the SIP 200 OK response to the network and then SHALL return to Step 5 to handle any subsequent NOTIFY received from the network.

The OITF SHALL ensure that the presence related data conforms to the appropriate XML schemas.

5.4.4.2 Procedure for Cancellation of a Subscription to Presence on the HNI-IGI interface

This procedure MAY be invoked at any time.

The OITF SHALL de-register the IPTV end user before invoking this procedure.

The procedure is essentially the same as the procedure for initiating a subscription to the Presence event, except that the X-OITF-Expires header in Table 55 SHALL be set to 0.

Table 55: List of HTTP extension headers for a SUBSCRIBE Request (OITF→IG)

X-OITF HTTP Header	Source of Information for Coding purposes
X-OITF-Request-Line Note: The request URI SHALL be set to the Public identity of the IPTV end user who has just registered	RFC 3261 [SIP] SUBSCRIBE <Request URI> SIP/2.0
X-OITF-From	RFC 3621 [SIP]
X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3621 [SIP]
X-OITF-Event	RFC 3265 [SIP-EVNT], OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**

X-OITF-Accept	RFC 3265 [SIP-EVNT], OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**
X-OITF-Contact Notes: 1. URI parameter MUST be included, and MUST match the Contact header included in the registration request. 2. Expires parameter SHOULD be included 3. Priority parameter SHOULD be included IG includes all other mandatory parameters that are absent.	RFC 3621 [SIP]
X-OITF-Call-ID	RFC 3621 [SIP]
X-OITF-CSeq	RFC 3621 [SIP]
X-OITF-Expires Note: If absent a default value shall be assumed by the IG	RFC 3621 [SIP]
X-OITF-Content-Type	RFC 3265 [SIP-EVNT], OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**
X-OITF-Content-Length	RFC 3261 [SIP]

Table 56: List of HTTP extension headers for the response to a SUBSCRIBE to Presence (IG→OITF)

X-OITF HTTP Header	**Source of Information for Coding purposes**
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Expires	RFC 3261 [SIP]
X-OITF-Contact	RFC 3621 [SIP]

Table 57: List of HTTP extension headers for a SIP NOTIFY (IG→OITF)

X-OITF HTTP Header	**Source of Coding Information**
X-OITF-Request-Line Note: The Request URI MUST match the contact URI included in the contact field of the SIP SUBSCRIBE	RFC 3621 [SIP] NOTIFY <Request URI> SIP/2.0
X-OITF-From	RFC 3621 [SIP]
X-OITF-To	RFC 3621 [SIP]
X-OITF-Event	RFC 3265 [SIP-EVNT], OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**
X-OITF-Call-ID	RFC 3621 [SIP]
X-OITF-Subscription-State	RFC 3265 [SIP-EVNT],OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**
X-OITF-CSeq	RFC 3621 [SIP]
X-OITF-Content-Type	RFC 3265 [SIP-EVNT] and OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**
X-OITF-Content-Length	RFC 3261 [SIP]
X-OITF-Contact	RFC 3621 [SIP]

Table 58: List of HTTP extension headers for a Response to a received SIP NOTIFY OITF→IG

X-OITF HTTP Header	Source of Coding Information
X-OITF-Response-Line	RFC 3621 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3621 [SIP]
X-OITF-To	RFC 3621 [SIP]
X-OITF-Call-ID	RFC 3621 [SIP]
X-OITF-CSeq	RFC 3621 [SIP]
X-OITF-Contact	RFC 3621 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]

NOTE: Cancellation of subscription is not required if the X-OITF-Expires header was set to 0 in the initial SUBSCRIBE request

5.4.4.3 Refreshing the Subscription to the Presence Event

It is the responsibility of the application (in the OITF) initiating the subscription procedure to refresh the subscription according to the refresh subscription timer received in the response during the subscription process. Refreshing SHOULD be performed before the expiry of the refresh timer. A subscription that is not refreshed before the expiration of the refresh subscription timer SHALL be terminated by the network.

The procedure for refreshing the subscription to the Presence event is the same as the procedure for subscribing to Presence.

5.4.4.4 Procedure for Publishing Presence information

This procedure for publishing an event MAY be invoked from either a DAE application or an embedded application in the OITF. The procedure is as follows:

Step 1: The OITF SHALL send an HTTP POST request to the IG over the HNI-IGI interface, as described in section 5.5.1, "OITF-IG Interface (HNI-IGI)." The content of the HTTP Request SHALL be as follows:

- **HTTP Request Header** including the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <list of SIP headers encoded as HTTP headers> - as per Table 59
- **HTTP Request Body:** As per section 5.4.4.6, "Presence Notification and Publish Schema."

Step 2: The IG SHALL validate that the request includes all the mandatory SIP headers for the publication process as per Table 59. The IG SHALL reject a request that is missing any mandatory SIP headers with a non-200 OK HTTP response, including the reason for rejection.

Step 3: The IG SHALL send a SIP PUBLISH to the network. When the IG receives the response, the IG SHALL return a 200 OK HTTP response (or other appropriate responses) to the OITF to report the response to the publish request. The response SHALL include a list of SIP headers as per Table 60, in addition to the normal HTTP headers as per RFC 2616 **[HTTP]**.

Table 59: List of HTTP extension headers for the PUBLISH Request (OITF→IG)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Request-Line Note: The request URI MUST be set to the IMS Public User Identity of the IPTV end user who has just registered	RFC 3261 [SIP] PUBLISH <Request URI> SIP/2.0
X-OITF-From	RFC 3261 [SIP]

X-OITF-To The URI part of X-OITF-To SHALL be set to the value of the Request URI in the "X-OITF-Request-Line"	RFC 3261 [SIP]
X-OITF-Event	RFC 3261 [SIP], OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-Expires	RFC 3261 [SIP], RFC 3903 [RFC3803]
X-OITF-SIP-If-Match	RFC 3903 [SIP]
X-OITF-Content-Type	RFC 3261 [SIP]
X-OITF-Content-Length	RFC 3261 [SIP]

Table 60: List of HTTP extension headers for a response to SIP PUBLISH (IG→OITF)

X-OITF HTTP Header	Source of Coding Information
X-OITF-Response-Line	RFC 3261 [SIP] SIP/2.0 <response>
X-OITF-From	RFC 3261 [SIP]
X-OITF-To	RFC 3261 [SIP]
X-OITF-Call-ID	RFC 3261 [SIP]
X-OITF-CSeq	RFC 3261 [SIP]
X-OITF-ETag	RFC 3261 [SIP], RFC 3903 [RFC3803]
X-OITF-Expires	RFC 3261 [SIP]

5.4.4.5 Procedure for Refreshing Published Presence information

It is the responsibility of the OITF to refresh the published presence information before the refresh timer expires. A published event that is not refreshed SHALL be deleted in accordance with RFC 3903 [RFC3803].

5.4.4.6 Presence Notification and Publish Schema

When the IPTV Presence service is active, the body of the PUBLISH request SHALL include the extended OMA presence schema compliant to section 5.1.6 of [TS183063] "Procedure for IPTV presence service".

In the XML document, each service is described by the "service-description" OMA parameter as specified in OMA-ERP-Presence_SIMPLE-V1_1-20080627-A **[SMPL-PRES]**. A new "service-id" is defined for IPTV with the following values:

- IPTV-BC: Scheduled Content service
- IPTV-CoD: Content on Demand Service
- IPTV-NPVR: Network PVR Service
- IPTV-Hybrid: DVB-T/H/C/S Service

When the user has an active IPTV Presence service, the <tuple> element pertaining to the active service SHALL contain the corresponding element as defined in the presence schema.

The presence schema extension for Open IPTV Forum Presence service is defined in 0, "Presence XML Schema".

5.5 Protocols System Infrastructure Functions

5.5.1 OITF-IG Interface (HNI-IGI)

5.5.1.1 HNI-IGI Message Types

The HTTP protocol is used to exchange information between the IG and the OITF. The IG behaves as an HTTP server and the OITF behaves as an HTTP client. The OITF part MAY be implemented either in native code or in DAE applications.

There are several aspects of information on the HNI-IGI interface.

- Normal HTTP headers
- Application specific information that is translated by the IG into SIP headers. These are included as HTTP extension headers and have the same name as in the SIP message, but are prefixed with X-OITF.
- Application specific information that forms the Body of a SIP message. This corresponds to the SIP message body and is included as a body in the HTTP request or response. An example message body type is SDP.
- HNI-IGI auxiliary information that is only used between OITF and IG. These parameters are appended with X-HNI-IGI. An example are those related to the fetching of GBA credentials by the OITF for re-use of GBA authentication mechanism for single sign-on.

The general format of an HNI-IGI HTTP request is

```
HTTP POST <IG URI>/<HNI-IGI message type>
<HTTP headers>
<X-OITF extension headers> or <X-HNI-IGI extension headers>
Content-Type: <…>
Content-Length: <Number>
<Message body>
```

The general format of an HNI-IGI HTTP response is

```
HTTP/1.1 <HTTP response>
<HTTP headers>
<X-OITF extension headers> or <X-HNI-IGI extension headers>
Content-Type: <…>
Content-Length: <Number>
<Message body>
```

The following table lists the HNI-IGI message types

Table 61: HNI-IGI Message Types

HNI-IGI message type	Meaning
PENDING_IG	The message is a pending HTTP request, that SHALL only be responded to by the IG when it needs to contact the OITF as a result of an incoming request from the network (e.g. an incoming MESSAGE)
SIP	The message is an HNI-IGI message corresponding to a SIP message. The IG must translate this into a corresponding SIP message by adding and changing the relevant headers.
AUX	The message is an HNI-IGI message that does not translate to a SIP message. The IG processes this message and responds accordingly.

Messages over the HNI-IGI interface can be sent in both directions.

- Normal HTTP requests are used for requests from the OITF and responses from the IG.
- There must be a HTTP request from the OITF to the IG with the response pending to allow new (unsolicited) messages from the network to be sent from the IG to the OITF in the response., This is a special kind of HNI-IGI message, called PENDING_IG

An example of SIP header that is mapped to an HTTP extension header is: “From: david@oiptv.org” that becomes “X-OITF-From:david@oiptv.org”

5.5.1.2 HNI-IGI messages in the OITF to IG direction

When the IG receives an HNI-IGI message, it SHALL add or change all SIP headers that are not specific to the application (Tags, Call ID, via, request URI etc.) while translating from HTTP to SIP.

The following table lists header values for the HNI-IGI protocol that the IG and OITF SHALL support and lists the IG action on those headers

Table 62: X-OITF HTTP Extension Headers and IG actions for OITF→IG messages

HNI-IGI Header	Description	IG Action
X-OITF-Request-Line	This is a special header which contains the SIP method and request URI for the corresponding SIP message, when the SIP message is a request, e.g. X-OITF-Request-Line PUBLISH sip:david@oiptv.org SIP/2.0	The IG SHALL map this field to the SIP request line.
X-OITF-Response-Line	This is a special header which contains the response line of the corresponding SIP message, when the SIP message is a response, e.g. 200 OK	The IG SHALL map this field to construct the SIP response line.
X-OITF-Call-ID	Keeps track of sessions and dialogs.	The IG SHALL use this field internally between an OITF and the IG to keep track of sessions. The IG replaces it with a value maintained in SIP state machine on the SIP side.
X-OITF-Contact	Each method has its own use of the Contact field.	The IG SHALL map to the corresponding SIP header. The IG MAY add other parameters.
X-OITF-CSeq	Used to keep track of requests and responses.	IG SHALL use this field internally between an OITF and the IG to keep track of requests and responses, and replace it with a value maintained by the IG on the SIP side. The IG SHALL include the same value in subsequent responses to the OITF. The OITF SHALL respond with an error code if the value is incorrect.
X-OITF-From		The IG SHALL map to the corresponding SIP header. The IG may add information in sub-fields.
X-OITF-Event		The IG SHALL map to the corresponding SIP header.
X-OITF-Expires		The IG SHALL map to the corresponding SIP header.
X-OITF-To		The IG SHALL map to the corresponding SIP header.

X-OITF-Content-Type		The IG SHALL map to the corresponding SIP header, and SHALL match it with the actual body included in the HTTP request.
X-OITF-Content-Length		The IG SHALL verify the length of the message and insert the value in the SIP message.
X-HNI-IGI-Request	This header specifies the request type of the HNI-IGI message.	See appropriate sections.

The above headers are not present in all HNI-IGI messages, and are not the only headers that can be present.

The OITF SHALL use an IMPU in X-OITF headers where an IMPU is required in the SIP header.

The OITF MAY include other headers that are application specific (e.g. X-OITF-Accept-Contact) in which case the IG SHALL include them transparently in the SIP method as long as they comply with the appropriate syntax for the header. Reference should be made to the various services using the HNI-IGI interface for a list of the headers that MUST be present.

5.5.1.3 HNI-IGI messages in the IG to OITF direction

When the IG translates a SIP message to an HNI-IGI HTTP message, it SHALL remove SIP Headers that should not be transmitted on the HNI-IGI interface, while translating from SIP to HTTP.

The following table lists header values in the SIP protocol that the IG and OITF SHALL support and the action the IG undertakes when mapping to the HNI-IGI protocol.

Table 63: Mapping of SIP header to X-OITF HTTP Extension Headers in IG→OITF

SIP header	Description	IG Action
Request Line (first line of SIP request message)	The Request Line contains the method and a SIP URI.	The IG SHALL use this field to construct the X-OITF-Request-Line
Response Line (first line of SIP response message)	The Response Line contains the response code and SIP version information.	The IG SHALL use this field to construct the X-OITF-Response-Line
Call-ID	Keeps track of sessions and dialogs.	The IG SHALL replace this with value used between IG and OITF in the X-OITF-Call-ID.
Contact		The IG SHALL map to the corresponding HNI-IGI header.
CSeq		The IG SHALL use this field to keep track of requests and responses on the HNI interface. The IG SHALL and replace it with a value maintained in the IG. The IG SHALL include the same value in subsequent responses to the OITF. The OITF shall respond with an error code if the value is smaller than the previous one.
From		The IG SHALL map to corresponding HNI-IGI header. The IG may add information in sub-fields.
Event		The IG SHALL map to the corresponding HNI-IGI header.
Expires		The IG SHALL map to the corresponding HNI-IGI header.
To		The IG SHALL map to the corresponding HNI-IGI header.

Content-Type		The IG SHALL map to the corresponding HNI-IGI header.
Content-Length		The IG SHALL verify the length of the message and insert value in the HNI-IGI message.

The above headers may not be present in all HNI-IGI messages.

The IG SHALL map any other received SIP headers by adding X-OITF- to the specific SIP header. Reference should be made to the various services using the HNI-IGI interface for a list of the headers that MUST be present.

The IG handles the SIP state machines.

5.5.1.4 HNI-IGI PENDING_IG Message

HNI-IGI PENDING_IG messages are sent by DAE and embedded applications in the OITF whenever these applications are ready to receive any incoming message from the network.

PENDING_IG messages MAY include a SIP Request or a SIP response. In this case, there is typically an ongoing SIP dialog between the OITF application and a peer SIP end-point in the IMS network.

HTTP headers included in a PENDING_IG message that includes a SIP request or a SIP response are the <X-OITF extension headers> that are pertinent to the application. The content of such a message SHALL be as follows:

- **HTTP Request Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-OITF Extension headers> Any number of those extension headers depending on the application
- **HTTP Request Body:** <SIP Message Body if applicable>

PENDING_IG messages that don't include a SIP request or a SIP response are typically sent by applications in the OITF that don't have any ongoing communication with a SIP peer but are prepared to handle incoming requests for the IPTV user associated with any active application running in the OITF, or applications that have an ongoing SIP dialog with a SIP peer and are prepared to receive any SIP messages within that dialog. The content of PENDING_IG messages that don't include a SIP request or a SIP response SHALL be as follows:

- **HTTP Request Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-OITF-Call-ID> Set to NULL for applications without an ongoing dialog or set to the proper value for applications with an ongoing SIP dialog.
  - <X-OITF-FROM> Set to the IMPU of the target user associated with any active application in the OITF for applications without an ongoing dialog. Not needed for applications with an ongoing SIP dialog
- **HTTP Request Body:** Empty

Note that once the target OITF application accepts an incoming request, the Call-ID in the outgoing response SHALL be set to the value used by the IG in its request to the OITF.

The content of the HTTP response to either of the above requests SHALL be as follows:

- **HTTP Response Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-OITF Extension headers> Any number of those extension headers depending on the application
- **HTTP Response Body:** <Appropriate Message Body if applicable>

5.5.1.4.1 Refreshing of HNI-IGI PENDING_IG Message

HNI-IGI PENDING _IG messages SHALL have to be refreshed periodically by the OITF. The refresh time SHALL be maintained in the IG and SHALL not exceed a SIP Session Expiry timer, or a SIP subscription Refresh timer for the SIP session under consideration.

To enable an OITF to refresh an HNI-IGI PENDING_IG request, the IG SHALL, upon timer expiry, send an HTTP 200 OK response that does not include any X-OITF-<Extension headers>.

Upon receipt of such a response, the OITF MAY decide to resend a new HNI-IGI PENDING_IG request or simply gracefully terminate the session.

5.5.1.4.2 Cancelling an HNI-IGI PENDING_IG Message

The IG considers an HNI-IGI PENDING_IG Request cancelled should it encounter one of the following events:

- The TCP connection on which the HNI-IGI PENDING_IG Request has been received is explicitly disconnected or timed out
- The IG received from the OITF application with an outstanding HNI-IGI PENDING_IG Request an HNI IGI SIP Request to terminate the session (a SIP BYE)
- The IG received from the OITF application with an outstanding HNI-IGI PENDING_IG Request an HNI IGI SIP Request to terminate an ongoing subscription (a SIP SUBSCRIBE (with X-OITF-Expiry set to 0)

For the last 2 cases, the IG SHALL send an HTTP 200 OK response that does not include any X-OITF-<Extension-headers> as a response to the PENDING-IG request. Optionally, the IG MAY empty its internal buffers that may include in-transit messages destined for the applications.

5.5.1.5 HNI-IGI SIP Message

HNI-IGI SIP messages are sent by DAE and embedded applications in the OITF whenever these applications are ready to send a SIP Request or a SIP response. The content of such a message SHALL be as follows:

- **HTTP Request Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-OITF Extension headers> Any number of those extension headers depending on the application
- **HTTP Request Body:** <SIP Message Body if applicable>

The content of the HTTP response SHALL be as follows:

- **HTTP Response Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-OITF Extension headers> Any number of those extension headers depending on the application
- **HTTP Response Body:** <SIP Message Body if applicable>

5.5.1.6 HNI-IGI Auxiliary Message

HNI-IGI auxiliary messages are sent by DAE and embedded applications in the OITF whenever these applications are ready to send messages that are neither SIP type nor PENDING_IG type (for example fetching GBA credentials)

The content of such a message SHALL be as follows:

- **HTTP Request Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-HNI-IGI-Request> - Identifies the request
  - <X-HNI-IGI Extension headers> Any number of those extension headers depending on the request

- **HTTP Request Body:** <Application Message Body if applicable >

The content of the HTTP response SHALL be as follows:

- **HTTP Response Header:** It includes the following:
  - <list of HTTP headers> - as per RFC 2616 **[HTTP]**
  - <X-HNI-IGI Extension headers> - Any number of those extension headers depending on the request
- **HTTP Response Body:** <Application Message body of applicable>

5.5.1.7 HNI-IGI Message Body

The HNI-IGI messages in either direction SHALL include transparently the appropriate SIP body for the different SIP methods in the HTTP message body.

5.5.1.8 Guidelines for Applications using the HNI-IGI interface

This section lists some guidelines that apply to DAE application and OITF applications that use the HNI-IGI interface. Both types of application are referred to as Application here:

- It is the responsibility of the Application to ensure that it provides all the SIP headers that are required for the correct operation of the application.
- The IG SHALL absorb “100 Trying” responses received from the network and not return them to the OITF.
- The IG SHALL transparently handle X-OITF-SIP headers received over the HNI-IGI interface unless specifically stated in this specification.
- It is the responsibility of the Application to generate an ACK as the 2XX final response of an INVITE transaction; for non-2XX responses, the IG SHALL generate the ACK.
- Validation (of message structure and XML schema) of received XML data SHALL be the responsibility of the Application.
- The Application SHOULD ensure that a SIP method is supported by the IG before using it.
- The IG SHALL return a 405 Method Not Allowed error fault if it receives a request over the HNI-IGI that includes a SIP method that it does not support.
- The Pending Request SHALL be refreshed as per section 5.5.1.4.1, “Refreshing of HNI-IGI PENDING_IG Message”.
- An HTTP pending request sent to the IG MAY include a SIP response (or SIP request if no response is expected) to be transmitted to the SIP network.
- The only identified case where a PENDING-IG request SHALL inlcude a SIP request is the case where the OITF application sends an ACK.

5.5.1.9 Error Recovery in the IG

This section covers the handling in the IG for encountered error-cases:

1) If the IG detects a timeout, or an explicit disconnection on all TCP links between an OITF application and the IG, the IG SHALL consider the application inactive. The application may become active again by re-establishing the TCP link with the IG. Any incoming SIP Messages destined for an application that is inactive SHALL result in an error message 487 Request Terminated (481 Call Leg/Transaction Does Not Exist is also an acceptable response if the IG concludes that the application is permanently inactive) being returned to the network as a response. For an inactive application, the SIP dialog in the IG shall eventually time-out and clear all resources in the IG and the network.

   In order to cater to the scenario of transient TCP link disconnects, and to allow incoming messages that are received by the IG during the time it takes the OITF to re-establish a TCP link following a disconnection, it is

recommended that the IG waits no more than 1 second before it responds to the network with a 487 or a 481 error message.

2) In case the GW is integrated into the IG (referred to as IG-GW), the IG-GW SHALL detect that an OITF is restarted upon receipt of an DHCP server discovery request (DHCPDISCOVER message) and IP address request (DHCPREQUEST message), and where the IG-GW internal state indicates that the OITF is powered on. In such a case, the IG-GW SHALL terminate all active SIP sessions, the IG-GW SHALL de-register all users that are logged in from the restarted OITF as stored in the IG-GW state. Note that the IG-GW is able to keep a mapping between the SIP dialogs ongoing, the IMPUs of registered users, and the IP addresses and DeviceIDs of the devices being used. Following deletion of stale SIP state and de-registration of users, the IG-GW SHALL act on the OITF start up high level procedure Requests.
   Note: this specification does not specify how error recovery works in case he GW is not integrated into the IG.

3) The OITF SHALL detect that an IG is restarted when all TCP links to the IG timeout simultaneously, or are explicitly disconnected. If this is the case, the OITF SHALL refrain from sending any message to the IG until such time that the OITF detects that the IG has restarted, using UPnP IG discovery procedure in that regard (polling). Following that, the OITF SHALL execute steps 2-6 of the “OITF Start up High-Level Procedure”.

For a transition period following an IG restart, active SIP sessions in the network (that will eventually timeout and be cleared) that are no longer active in the IG, will continue to send SIP messages to the IG, to which the IG SHALL respond with error message 481 Call Leg/Transaction Does Not Exist.

6 SIP and SIP/SDP

6.1 SIP/SDP Reference Points

This section defines the protocol for the use of SIP and SIP/SDP over the following reference points:

- NPI-19
- NPI-26
- NPI-30
- NPI-25
- NPI-3
- NPI-4
- UNIS-8

6.2 Protocols for IPTV Service Functions

6.2.1 Scheduled Content Service

The IG SHALL support the procedures specified in **[TS124503]** for originating sessions.

6.2.1.1 Protocol over UNIS-8

6.2.1.1.1 Session Initiation and Modification

Upon receiving a request from the OITF for the initiation of a Scheduled Content session (see section 5.2.1.1.1, "Session Initiation"), the IG SHALL generate an initial INVITE request as specified in **[TS124503]** for originating sessions.

The IG SHALL forward any received SIP response to the OITF including the information in the SDP

If the IG receives a 488 error code with warning 370 Insufficient Bandwidth, the IG SHALL send an error message to the OITF.

Session modification procedure is handled by the IG in the same way as a session initiation. See section 5.2.1.1.2, "Session Modification."

6.2.1.1.2 Session Termination

On receiving a request from the OITF for the termination of a Scheduled Content session (see section 5.2.1.1.3, "Session Termination"), the IG SHALL generate a BYE request as specified in **[TS124503]** for originating sessions.

Alternatively, on receipt of a BYE request from the IPTV Control FE, the IG SHALL forward the request to the OITF as a response to a PENDING_IG request (see section 5.5.1, "OITF-IG Interface (HNI-IGI)"). The behaviour of the UNIS-8 part of the IG SHALL comply with the procedure specified in **[TS124503]** for terminating UA.

6.2.1.2 Protocol over NPI-4

6.2.1.2.1 Session initiation

The IPTV Control FE SHALL support the procedures specified in **[TS183063]** section 5.3.1.1.

The IPTV Control FE SHALL support the procedures specified in **[TS124503]** that are applicable to an AS acting as a terminating SIP UA.

Upon receipt of a SIP INVITE request, the IPTV Control FE shall examine the request-URI to determine that it is a BC session initiation request. The IPTV Control FE SHALL use the IPTV Subscription Profile to check the service rights

for the requested broadcast service packages and multicast addresses. The IPTV Control FE shall examine the SDP offer parameters, as defined in **[TS183063]** section 5.3.1.1.

If the SDP parameters are validated successfully, the IPTV Control FE SHALL respond as defined in **[TS183063]** section 5.3.1.1.

If no bc_service_package attributes are included in the SDP offer, the IPTV Control FE SHALL include in the SDP answer one or more a=bc_service_package attributes, except if it knows that the RACS is or shall be pre-provisioned with the list of subscribed channels and if all the subscribed channels are allowed for the session. In this case, the inclusion of a=bc_service_package is OPTIONAL.

The service packages shall be populated according to the IPTV Subscription Profile to indicate the service packages and BC services.

6.2.1.2.2 Session modification

The IPTV Control FE SHALL support the procedures specified in **[TS183063]** section 5.3.1.2. Network initiated Scheduled content (BC) session modification does not apply.

6.2.1.2.3 Session termination

The IPTV Control FE SHALL support the procedures specified in [TS183063] section 5.3.1.4

6.2.2 Content on Demand

6.2.2.1 Retrieving missing parameters in the SDP prior to session setup using SIP OPTIONS

6.2.2.1.1 Protocol over UNIS-8

When a request to send a SIP OPTIONS is received from the OITF, the IG SHALL use the mapping specified in section 5.2.2.1.1, “Retrieval of Session Parameters.”

When the final response to the SIP OPTIONS message is received from the network as a SIP 200 OK including the RTSP SDP, the IG SHALL forward this information to the OITF

The information required in the returned SDP to complete the missing parameters in the SDP offer is:

- FEC Information including bandwidth for FEC streams,
- Transport protocol

6.2.2.1.2 Protocol over NPI-4, NPI-19, NPI-26

The OPTIONS message SHALL conform to **[TS124503]** and SHALL be forwarded through the ASM, IPTV Control and CDN Controller FE to the appropriate Cluster Controller, in the same way as for the INVITE message.

In certain cases, the CDN Controller MAY forward the SIP OPTIONS message to a default Cluster Controller.

On receiving the SIP OPTIONS message, the Cluster Controller SHALL issue an RTSP DESCRIBE to the CDF. In certain cases, the Cluster Controller MAY issue an RTSP DESCRIBE to a default CDF. The Content-type header of DESCRIBE message SHALL be set to “application/sdp”.

The SDP body included in the RTSP 200 OK response received from the CDF SHALL be included by the Cluster Controller in a SIP 200 OK response to the OPTIONS message. The SIP 200 OK message SHALL be forwarded all the way back to the IG.

This is the only case for which an OPTIONS message will be sent to a Cluster Controller.

Note: If, in a future release, other reasons warrant that the Cluster Controller receive the OPTIONS message, then support for discrimination between the various reasons for sending the OPTION will be required.

6.2.2.2 Procedure for Unicast Service Session Initiation

6.2.2.2.1 Session Initiation

The IG SHALL support the procedures specified in **[TS124503]** for initiating unicast sessions.

On receiving a request for a unicast session initiation from the OITF, the IG shall generate an initial INVITE request as specified in **[TS124503]** (for an originating UA). See section 5.2.2.1.2, “Session Initiation.”

See example messages in section C.1.1, “Example Messages for CoD session setup in a Managed Network.”

6.2.2.2.2 Protocol over NPI-4

The IPTV Control Function SHALL support the procedures specified in **[TS124503]** as applicable to an AS acting as a SIP proxy or B2B UA.

When receiving any SIP request, the IPTV Control FE SHALL examine the request to see if it is compatible with the user's subscription profile (e.g. parental control level). If the user is not allowed to initiate a session for the requested content, the IPTV Control FE SHALL reply with an appropriate SIP error response. If the user is allowed to initiate the session, the IPTV Control FE SHALL forward the SIP INVITE to a default CDN Controller.

The IPTV Control Function SHALL not change the user-part of the To header in order to retain the content-id in the INVITE request.

6.2.2.2.3 Protocol over NPI-19

The CDN Controller FE SHALL support the procedures specified in **[TS124503]** as applicable to an AS acting as a SIP proxy or B2B UA.

When receiving the SIP INVITE from the IPTV Control FE via the Authentication and Session Management FE through the NPI-19 reference point, the CDN Controller SHALL check the CoD content id in the user part of the “To:” header as well as the “From:” and “Via:” fields to determine the most appropriate Cluster Controller FE to serve the User's request.

Once the appropriate Cluster Controller FE is selected, the Content Delivery Network Controller FE SHALL forward the SIP INVITE to it by changing the “Request-URI” accordingly.

The CDN Controller SHALL NOT forward 301 or 302 responses from the Cluster Controller to the IPTV Control Function. The CDN Controller SHALL take one of the following actions on receiving a 301 or 302 response from the Cluster Controller:-

- Cancel the transaction
- Forward to another Cluster Controller
- Forward to the suggested CC as indicated in the 301/302 response
- Forward to another CDN Controller

6.2.2.2.4 Protocol over NPI-25

On receiving the request from the IPTV Control Function, the CDN Controller MAY decide to forward the request to another CDN Controller. In this case it changes the “Request-URI” accordingly.

6.2.2.2.5 Protocol over NPI-26

The Cluster Controller FE SHALL support the procedures specified in **[TS124503]** as applicable to a terminating UA.

When receiving a CoD session initiation SIP request from the CDN Controller, the Cluster Controller SHALL examine the CoD content identifier present in the user-part of the “To:” header and the media parameters in the received SDP offer and then choose the CDF.

If the requested content is not managed by this Cluster Controller, the Cluster Controller SHALL return a 301 response, or a 302 response for any other reasons (e.g. load-balancing) The Cluster Controller MAY indicate one or more Cluster Controller addresses in the contact header as indicated in RFC 3261 **[SIP]**.

If the request is not acceptable to the Cluster Controller, it SHALL reply with an appropriate SIP error response.

The Cluster Controller SHALL reply with an appropriate SIP error response if the request is acceptable to the Cluster Controller but none of the Content Delivery Functions can handle the offer.

If the request is acceptable to the Cluster Controller and a CDF can handle the request, the Cluster Controller SHALL initiate an RTSP session using the RTSP SETUP message to the chosen CDF to determine its server ports and the RTSP session ID.

Following the successful conclusion of the RTSP session setup, the Cluster Controller allocates an RTSP server port, binds it to the CDF RTSP server port and answers with a SIP 200 OK, including the SDP answer.

The SDP parameters for the RTSP channel SHALL be set as follows:-

- An m-line for an RTSP stream with the format: m=<media> <port> <transport> <fmt>
  (ex. `m=application 554 tcp iptv_rtsp`)
  - The <media> field SHALL have a value of “application”.
  - The <port> field SHALL be setup according to RFC 4145 **[SDP-TCP]**. The port number SHALL be set to the port allocated by the Cluster Controller.
  - The <transport> field SHALL be identical to the one received in the SDP offer in the initial INVITE.
  - The <fmt> field SHALL be identical to the one received in the SDP offer in the initial INVITE.
- A c-line SHALL include the network type with the value set to “IN”, the address type set to “IP4” and the IP address for the RTSP commands.
  (ex: `c=IN IP4 <RTSP IP address>`)
- An “a=setup” attribute SHALL be present and set to “passive”, indicating that the connection is initiated by the other endpoint (OITF), as defined in RFC 4145 [SDP-TCP]. (ex: `a=setup:passive`)
- An “a= connection” attribute SHALL be present and set to “new” as defined in RFC 4145 [SDP-TCP].
  (ex: `a=connection:new`)
- One or more a=fmtp lines representing RTSP specific attributes set as follows
  - A “fmtp:iptv_rtsp h-uri” attribute SHALL be set to the RTSP URI of the Cluster Controller to be used in the RTSP requests. The h-uri can be in the form of an absolute or relative URI. If an absolute URI is specified then it SHALL be used in subsequent RTSP requests. If a relative URI is specified in the form of a media path, then the RTSP absolute URI could be constructed by the OITF using the IP Address (from c-line) and port (from m-line) as the base followed by h-uri value for the media path.
    (i.e. fmtp:iptv_rtsp h-uri=<request-uri>)

    An absolute URI SHALL have precedence over a c-line if the latter is provided.
  - The Cluster Controller SHALL include a “fmtp:iptv_rtsp h-session” attribute representing the session-id of the RTSP session to be used by the OITF during media control. Optionally, a timeout parameter MAY be specified with a numeric timeout interval in seconds for keep-alive (refer to section 7.1.1.2.3, “RTSP Control for media delivery.”) If the timeout parameter is not specified, then a default value of 60 seconds SHALL be used (refer to section 12.37 in [RTSP].)
    (i.e. a=fmtp:iptv_rtsp h-session=<rtsp-session>[; timeout=<timeout>])

    Note that if both RTP and RTCP are used in the session, the RTSP server (CDF) can use the received RTCP messages as an indication that the OITF is still connected to the session. This avoids requiring explicit RTSP keep-alive signalling. The RTSP server can easily associate the RTCP messages to the RTSP Session-ID using the RTCP message transport address and the SSRC of the media source. If this method is used and no RTCP messages are received after the default timeout period, the RTSP server MAY tear down the session. Details of this methodology are explained below in the b=RR:<bandwidth-value> line.

- An m-line for the actual content which indicates the type of the media, the transport protocol and the port of the related content delivery channel from the response message for the RTSP DESCRIBE. If a fmt parameter is in the SDP offer it SHALL be completed with the supported format by the CDF,

- A c-line SHALL include the network type with the value set to IN, the address type set to IP4 and the unicast address of the stream related to the content delivery channel.
(i.e. `c=IN IP4 <IP_ADDRESS>`)

- A b-line SHALL contain the proposed "session bandwidth" for the COD media stream. Note that this bandwidth value includes the IP and UDP headers (see section 6.2 of **[RTP]**).
(ex. `b=AS:64`, indicating 64kbps)

- An a-line with a "sendonly"
(ex. `a=sendonly`)

- A b-line, b=RR:<bandwidth-value>, specifying the agreed bandwidth value (in kbps) the OITF SHALL allocate for sending Receiver Reports (RR) in the COD session.

    - The Cluster Controller MAY set the bandwidth value in the answer to zero (0) even if a non-zero value is requested by the OITF. In the event a non-zero value is requested by the OITF, the Cluster Controller SHALL NOT change the proposed bandwidth value to a different non-zero value as this could force the OITF to use a bandwidth value it may not be able to allocate causing COD session failure.

- Optionally, a b-line b=RS:<bandwidth-value> specifying the amount of bandwidth (in kbps) that the COD session senders (in this case only one COD server) SHALL allocate for RTCP Sender Reports (SR). This value MAY be set to zero (0), b=RS:0. Setting this value to zero (0) is not recommended if several streams will be synchronised.

6.2.2.3 Session Termination

Session termination for COD SHALL follows the same SIP procedures as session termination for the Scheduled Content service. See section 6.2.1.1.2, "Session Termination."

6.3 Protocols for Service Access and Control Functions

6.3.1 Service Provider discovery

6.3.1.1 Protocol for UNIS-8 and NPI-30

The IPTV Service Provider Discovery FE SHALL generate and/or provide the Service Provider Discovery information.

The IG SHALL follow the following procedure to retrieve Service Provider Discovery information:

Step 1: The IG SHALL send a SIP SUBSCRIBE to the network, to subscribe to the "ua-profile" event, and SHALL wait for the response to the subscription request.

Step 2: When a SIP NOTIFY is received by the IG for a "ua-profile" event, the IG SHALL store the body of the SIP NOTIFY.

Step3a: If the IG receives a HTTP GET for the Service Provider information, it SHALL return the body of the SIP NOTIFY (from step 2) in the HTTP response body.

Step 3b: If the IG receives a HTTP POST on the HNI-IGI interface from the OITF which includes a SIP SUBSCRIBE with a message body associated with the appid "urn:oipf:application:iptv-SP-discovery", the IG SHALL send a SUBSCRIBE to the network with the following capabilities:

The message body SHALL include what was received from the OITF, which are the capabilities of the OITF which are sent to the Service Provider Discovery FE. The details of the SIP SUBSCRIBE are as specified in [TS183063] section 5.1.2.2.1. To wit:

- The Content Type header SHALL be set to "application/vnd.oipf.ueprofile+xml"

- The UserEquipmentID is a unique global identifier of the device.

- The User Equipment Class SHALL take values “TV-OITF”, “STB-OITF”, according to the implementation options in Release 1 Architecture specification **[ARCH]** Annex D.

When the Service Provider Discovery FE receives a SUBSCRIBE request, it MAY check the user’s IPTV subscription profile and provide a personalized Service Provider Discovery information to the OITF. Filtering may also be performed if device capabilities are available to the SDF.

If the Service Provider Discovery FE receives a SIP SUBSCRIBE message body from the IG carrying UE capabilities, the Service Provider Discovery FE shall process the SIP request as specified below.

On successful subscription, the Service Provider Discovery FE SHALL generate a 200 OK response. The Service Provider Discovery FE SHALL then send a NOTIFY request to the OITF in accordance with RFC 3265 [SIP-EVNT].

The contents of the SIP NOTIFY request SHALL be as follows:

- Extend the existing “ua-profile” event package for SIP NOTIFY as follows:
  - The Event header SHALL be set to the “ua-profile” event package.
  - The “effective-by” parameter for the event header SHALL be set to 0.
  - The content type SHALL be set to “application/vnd.oipf.spdiscovery+xml”.

The Service Provider Discovery Information SHALL be delivered in the message boday and SHALL conform to the schema defined in **[META]**.

Note: If the above extension is not accepted by the IETF, then the use of a new method (**New Event package)** should be re-examined. (See 0, “New Event package for SIP SUBSCRIBE /NOTIFY (informative).”)

When the IPTV Service Provider Discovery FE knows of a change to the Service Discovery, Service Provider or Selection Information, the IPTV Service Provider Discovery FE SHALL inform the OITF of this change by sending a SIP NOTIFY message.

6.3.2 User Registration and Network Authentication

6.3.2.1 User Identity Modelling

Every IMS Subscription SHALL be allocated a single unique default IMS Pubic Identity by the Service Platform Provider. This SHALL be the identity that is registered in the IMS domain when an OITF is turned on.

Every IPTV end-user in an IMS Subscription MAY be associated with an IMS Public User Identity by the Service Platform Provider.

This release complies with option 1 in section D.4 of **[ARCH]**.

6.3.2.2 Procedure for User Registration and Authentication in a Managed Model on UNIS-8

The following SHALL be supported by the IG on the UNIS-8 interface for user registration:

- The IG SHALL support the 3GPP IMS registration procedure as per ETSI TS 124 503 **[TS124503]**. This includes handling of user authentication and authorization. This procedure shall be invoked when the IG powers up (in this case the default identity SHALL be registeres) or upon receipt of an HTTP POST from the OITF with the REGISTER method.
- The IG SHALL report to the OITF the final outcome of any OITF-initiated registration or de-registration.
- The IG SHALL be stateful for all successful registrations until de-registration occurs.
- For IG-initiated registration procedures, the IG is responsible for refreshing the registration before the registration expiry time.

The following SHALL be supported by the IG on the UNIS-8 interface for user de-registration:

- The IG SHALL support the 3GPP IMS de-registration procedure as per ETSI TS 124 503 **[TS124503]**. This procedure shall be invoked upon receipt of an HTTP POST from the OITF with the REGISTER method and when the IG shuts down.

The following SHALL be supported in the IG on UNIS-8 for subscription to the Registration event:

- For OITF-initiated registrations, the IG SHALL support subscription to the registration-state event package as per ETSI TS 124 503 **[TS124503]**.
- For IG-initiated registrations, and following a successful registration process, the IG SHALL SUBSCRIBE to the registration event package in accordance with ETSI TS 124 503 **[TS124503]**.
- On request from the OITF, as well as for IG-initiated registrations, the IG SHALL refresh the registration-state event package subscription in accordance with ETSI TS 124 503 **[TS124503]**.
- For OITF-initiated registrations, the IG SHALL NOT store any registration event related data, but SHALL be stateful of the subscription. For IG-initiated registrations, the IG SHALL store registration event related data.
- For OITF-inititaed registrations, the IG SHALL support terminating a subscription to the registration-state event package as per ETSI TS 124 503 **[TS124503]**.
- For IG-initiated deregistration, and following the successful deregistration process, the IG SHALL terminate the subscription to the registration event package, as per ETSI TS 124 503 **[TS124503]**.

The appropriate procedure (SIP Digest or IMS AKA) SHALL be follwed by the IG for user registration and authentication.

6.3.3 Notification of Service Profile changes

6.3.3.1 Notification of Service Profile changes Protocol on UNIS-8

6.3.3.1.1 Subscription to Notifications of Service Profile changes

If subscription to notification of changes is requested by the OITF, the IG SHALL send a SUBSCRIBE request to the IPTV Service Profile FE in accordance with IETF draft-ietf-sip-xcapevent-03 **[XCAP-EVT]** and draft-ietf-simple-xcap-diff-09.txt **[XCAP-DFF]**.

The IG will process the request from the OITF and will generate a SUBSCRIBE request, that SHALL be as specified in **[TS183063]** section 5.1.5.1.

A well known PSI mechanism SHALL be used in the request URI of the SUBSCRIBE request.

Note: For changes that apply to a very large number of subscribers, it is up to Service Provider to set up proper rules in the ‘notifier function’ to make the notification procedure scalable.

6.3.3.1.2 Processing of notifications

Refer to **[TS183063]** section 5.1.5.2

6.4 Protocols for Communication Services

6.4.1 CallerID

6.4.1.1 Procedures for Caller ID on UNIS-8

6.4.1.1.1 Instant Message based Caller ID

Instant Message based Caller ID is identical to Instant Messaging where the incoming message includes the caller id. For further details reference should be made to section 6.4.2.1, “Procedure for Instant Messaging on UNIS-8.”

6.4.1.1.2 IMS telephony service based caller identification [OPTIONAL]

IMS telephony service based caller identification is based on the reception of the regular SIP session INVITE request. The incoming session request message includes the caller identification.

Support of this feature by the IG is OPTIONAL.

6.4.2 Instant Messaging

6.4.2.1 Procedure for Instant Messaging on UNIS-8

Instant Messaging complies with the page mode of operation. The following SHALL be supported by the IG

- Incoming SIP MESSAGE messages to the IG MUST conform to OMA Instant Messaging using SIMPLE Draft OMA-TS-SIMPLE_IM-V1_0-20080820-D **[SMPL-IM]** to be acceptable for processing by the IG. Non-conformant SIP MESSAGE messages SHALL be rejected in accordance with **[SMPL-IM]** with the appropriate response code.
- For an incoming SIP MESSAGE to the IG that is conformant to **[SMPL-IM]**, the IG SHALL forward the message to the OITF using HNI-IGI Notification procedure and SHALL send a 202 Accepted to the originating end.
- An outgoing SIP MESSAGE SHALL conform to **[SMPL-IM]**. The response to the SIP MESSAGE SHALL comply with **[SMPL-IM]**.
- The IG SHALL NOT retain any state information once the transaction is completed.

6.4.3 Presence

6.4.3.1 Procedure for Presence on UNIS-8

The following SHALL be supported by the IG for subscription to Presence:

- An outgoing SUBSCRIBE message for a subscription to the Presence event, or cancellation of an existing subscription SHALL comply with **[SMPL-PRES]**.
- An incoming NOTIFY messages that does not comply with **[SMPL-PRES]** SHALL be rejected with the appropriate error code in accordance with **[SMPL-PRES]**, and no further processing shall be performed.
- On request from the OITF, the IG SHALL refresh the Presence subscription in accordance with **[SMPL-PRES]**.
- The IG SHALL not store any presence related data, but SHALL be stateful to the subscription.
- The IG SHALL consider a subscription terminated if it is not renewed by the OITF.

6.4.3.2 Procedure for Publishing Presence Information on UNIS-8

When requested by the OITF, the IG SHALL support the SIP PUBLISH request and response in accordance with **[SMPL-PRES]** for publishing presence information.

6.4.4 Chatting

6.4.4.1 IM Session using MSRP over UNIS–8

The IG SHALL conform to the Client Procedure as described in OMA-TS-SIMPLE_IM-V1_0-20080820-D **[SMPL-IM]**.

The IG SHALL perform path mapping between Chatting peers as indicated in section 5.4.3, “IM Session (Chat using MSRP).”

The IG SHALL handle translation of Chat session initiation and teardown procedures when requested by the OITF as per section 5.3.5, “Remote Management.”

The IG SHALL handle translation of outgoing and incoming MSRP chat message as per section 5.4.3, “IM Session (Chat using MSRP).”

6.4.4.2 IM Session using MSRP over NPI–3

The P2P Chatting communication enablers FE shall confirm to the IM Server procedures described in OMA-TS-SIMPLE_IM-V1_0-20080820-D **[SMPL-IM]**.

7 RTSP

This section defines the protocol for the use of RTSP over the following reference points:

- UNIS-11
- NPI-10

7.1 Protocols for IPTV Service Functions

7.1.1 Use of RTSP for CoD

7.1.1.1 RTSP Profile for the unmanaged model over UNIS-11 and NPI-10

The RTSP protocol SHALL be used on UNIS-11 and NPI-10 for unicast service setup and delivery. The OITF SHALL obtain an RTSP request URL from the content guide, prior to delivery of the media from a Cluster Controller. The use of RTSP SHALL comply with RFC 2326 [RTSP] and with the following profile.

The following describes the RTSP Profile for this Release. The functionalities not identified in this section are out of scope of OIPF:

- NPT Range time format SHALL be supported by clients and servers, section 3.6 of **[RTSP]**
- The extension mechanism using option tags in section 3.8 of **[RTSP]** SHALL NOT be used
- Since in this Release is constraint to one media stream per session (one m= line of audio/video data), then:
    - This profile is not multi-server capable (section 1.4 of **[RTSP]**)
    - This profiled does not support aggregate control. Aggregated support allows to control several associated streams (e.g., video and sound track) using one RTSP URI.
    - This profile does not support “pipelining” of RTSP messages, see S. 9.1 Pipelining
- This profile SHALL support the following MIME types:
    - SDP (application/sdp) as the normative session description format, and
    - The MIME type text/parameters for GET_PARAMETER
- Servers SHALL NOT use RTSP proxy features
- Servers SHALL NOT use encryption or authentication
- The RTSP client SHALL understand the class of each status code, i.e., 1xx Information, 2xx Success, 3xx Redirection, 4xx Client Error, and 5xx Server Error. Additionally, a client SHALL understand the following status codes:
    - “200” ; OK
    - “301” ; Moved Permanently
    - “302” ; Moved Temporarily
    - “400” ; Bad Request
    - “404” ; Not Found
    - “405” ; Method Not Allowed
    - “408” ; Request Time-out
    - “500” ; Internal Server Error

- "501" ; Not Implemented

- Servers SHALL NOT use interleaved RTP and RTSP over TCP, as per section 10.12, "Embedded (Interleaved) Binary Data" of **[RTSP]**.

- Clients SHALL NOT use PLAY messages in the PLAY state as keep-alives (section 10.5 of **[RTSP]**).

- Servers SHALL NOT use RTSP over UDP, see Appendix G of **[RTSP]** and related functionality like the rtspu URI scheme in section 22.14.3 of **[RTSP]**.

- Regarding the RTSP Header definitions, the client SHALL support the following headers:

  - Accept, Allow, Content-Type, CSeq, Location, Public, Range, RTP-Info, Scale, Session, Transport

Additionally, the following clarifications and best practices are added:

- RTSP URIs SHALL follow encodings and escape conventions as per [RFC3986]. This is an RFC reference update. This RFC defines a fragment part to the RTSP URI, which was hinted but not specified in **[RTSP]**.

- Regarding section 9, "Connections" of **[RTSP]**: it is RECOMMENDED that servers use persistent TCP connections. This reduces session management complexity (keep-alive, tear down, etc) in the clients and the servers. Clients SHOULD use persistent connections.

- Regarding keep-alive mechanisms, the following mechanisms are RECOMMENDED in this order:

  - If the OITF has agreed to send RTCP packets, it is RECOMMENDED that these be re-used to keep the RTSP session alive.

  - If not, it is RECOMMENDED that the same empty RTP packets with payload type value 20 used for NAT traversal (see G.6.2, "NAT Traversal and keep-alive messages for CoD") be re-used to keep the RTSP session alive.

  - Otherwise, the SET_PARAMETER Method (or GET_PARAMETER, whichever is supported) with an empty body SHOULD be uased.

  - Finally, the RTSP OPTIONS Method SHOULD be used

- Regarding section 12.17, "CSeq" of **[RTSP]**, the current best practice rules and clarifications are added were not clear in RTSP, in particular:

  - If a server returns a "400 Bad Request" response because the client did not include a CSeq header, then the response SHALL not include a CSeq header.

  - The CSeq header value MUST be increased by one for each new RTSP request

  - It is RECOMMENDED that CSeq value starts at "1" (one)

- Regarding non-seekable streams, Annex C.1.5: these are indicated by using open-ended time intervals via a=range attribute in SDP. E.g., a=range:npt=0-

- Regarding Content-Length: an RTSP message with a message body SHALL include the Content-Length header. This can be misinterpreted from sections 4.3 and 12.14 of **[RTSP]**, because section 4.3 of **[RTSP]** refers to HTTP/1.1 (which only recommends it) while section 12.14 of **[RTSP]** clearly says that it MUST be included.

- Regarding RTP-Info, section 12.33 of **[RTSP]**:

  - The client SHALL NOT use the RTP SSRC parameter ("ssrc:") in a SETUP request. This is deprecated because it incompatible with the specification of RTP in RFC 3550 **[RTP]**.

  - The URL values in the "url=" parameter SHALL be quoted, e.g.: RTP-Info: url="rtsp://live.example.com/concert/audio". This allows the URL to contain ";" and "," characters.

  - If the value of the "url" parameter in RTP-Info header is a relative URI, then the Request-URI is used as base-URI. If it is an absolute URI, then this URI is the same as in the SETUP message.

7.1.1.1.1 RTSP Session Setup

When performing RTSP session setup, the OITF SHALL use the request URL to send a DESCRIBE message to the Cluster Controller to obtain a media SDP. The DESCRIBE message SHALL include the Accept header with the application/sdp content type. The Cluster Controller or CDF SHALL return a Content-Type header with application/sdp and the format of the body SHALL be according to RFC 4566 **[SDP]**. The OITF SHALL then issue a SETUP request to the Cluster Controller.

If the Cluster Controller can handle the request, the DESCRIBE and SETUP messages SHALL be forwarded to the most appropriate CDF.

If the Cluster Controller cannot handle the request, the Cluster Controller SHALL reply with a redirect response (Moved) message containing a URL of another Cluster Controller. The redirection MAY occur when receiving either a DESCRIBE or a SETUP.

If the setup is successful the CDF SHALL reply with a 200 OK message that SHALL be proxied by the Cluster Controller to the OITF. After receiving the setup response the OITF MAY send PLAY and PAUSE messages to the Cluster Controller.

The Cluster Controller SHALL modify the RTSP URL to forward the RTSP messages to the chosen CDF function, when the messages are initiated from the OITF.

When receiving error messages from the CDF, the Cluster Controller SHALL either forward them to the OITF or try another CDF.

When the OITF receives an error message, it MAY display appropriate messages to the end user. The error messages MAY also be handled by the downloaded DAE or native application before being displayed to the user.

7.1.1.1.2 RTSP Control for media delivery

Handling of Media Control for Starting Playback

On receiving a request from the user to start playback, the OITF SHALL send an RTSP PLAY message to the Cluster Controller.

The RTSP fields in the RTSP PLAY message SHALL be filled as follows:

On receiving a request from the user to modify the playback, the OITF SHALL send an RTSP PLAY message with a request to modify the position, speed and/or direction of playback. The OITF indicates the direction and/or speed of playback by including a Scale header or changes the position of playback by including a Range header.

- The Scale header SHALL be set as follows:
  - 1 indicates normal play;
  - If not 1, the value corresponds to the rate with respect to normal viewing rate;
  - A negative value indicates reverse direction.

If the request is to pause the playback, the OITF SHALL send an RTSP PAUSE message.

On receipt of a RTSP PLAY or PAUSE request, the Cluster Controller SHALL forward the message to the chosen CDF.

The CDF SHALL respond with a 200 OK message. The contents of the 200 OK response SHALL be as follows:

- CSeq SHALL be set to the same value as that in the request.

Handling of Media Control for Retrieving Playback Information

For OITF devices that require retrieving the position and the duration parameter from the server for operational reasons, the OITF SHALL support the method GET_PAREMETER message for that purpose.

All OITF devices SHALL support the retrieval from the server of the scales parameter through the GET_PARAMETER message.

Any other parameter not supported by the Cluster Controller used in the GET_PARAMETER request SHALL be rejected by the Cluster Controller with an appropriate error code. An empty body SHALL be allowed for the RTSP keep-alive message.

If RTSP is used as a keep-alive, then the timeout for sending the request is based on the timeout parameter specified in the session header in the RTSP SETUP response. If timeout parameter is not specified then a default value of 60 seconds shall be used.

On receipt of the RTSP GET_PARAMETER request, the Cluster Controller SHALL forward the message to the chosen CDF.

The CDF SHALL respond with a 200 OK message with the requested values or empty in the case of a keep-alive message. The message SHALL be forwarded to the OITF.

Handling of Beginning and End of Stream

On receipt of the beginning-of-stream or end-of-stream indication from the CDF, the Cluster Controller MAY send an RTSP ANNOUNCE to the OITF with an indication that the beginning-of-stream or end-of-stream has been reached. The Notice header SHALL be included with the notice code value set to “2104 Start-of-Stream Reached” or “2101 End-of-Stream Reached”.

Note: The header and code is based on **[RTSP2-AN]**.

On receipt of the RTSP ANNOUNCE with an end-of-stream or beginning-of-stream indication, the OITF MAY take relevant actions to handle the event (e.g. terminating the session, rewinding the media stream, etc.). The OITF SHALL respond with a RTSP 200 OK.

Handling by the CDF for multiple PLAY messages in Play state

The CDF SHALL not queue successive PLAY messages for processing while in a play state. An incoming new PLAY message SHALL result in an immediate termination by the CDF of the processing associated with a previous pending PLAY message if applicable.

7.1.1.1.3 RTSP Session Teardown

To tear down a unicast session, the OITF SHALL use a RTSP TEARDOWN message and SHALL wait for a 200 OK response from the Cluster Controller. The Cluster Controller SHALL relay the RTSP TEARDOWN message to the CDF and relay the 200 OK message to the OITF.

7.1.1.1.4 Supported RTSP Messages

The OITF acting as an RTSP Client and the Cluster Controller acting as an RTSP Server SHALL support at least the following messages: RTSP SETUP, RTSP TEARDOWN, RTSP DESCRIBE, RTSP PLAY, RTSP PAUSE, RTSP GET_PARAMETER, RTSP ANNOUNCE, and RTSP OPTIONS.

The CDF as an RTSP server SHALL support at least the following messages: RTSP SETUP, RTSP TEARDOWN, RTSP DESCRIBE, RTSP PLAY, RTSP PAUSE, RTSP OPTIONS, and RTSP GET_PARAMETER.

7.1.1.2 RTSP for managed model over UNIS -11 and NPI-10

The RTSP protocol SHALL be used on NPI-10 for CoD session setup and UNIS-11 and NPI-10 for CoD service delivery.

In the managed model, the OITF SHALL obtain the appropriate RTSP request URI, RTSP session ID, and the RTP media parameters, prior to content delivery from the assigned Cluster Controller.

If the OITF supports RTSP/RTCP monitoring, it SHALL also include the a=OIPF-QoS-Metrics line, the a=rtcp-xr line and the b=RR line prior to content delivery from the assigned CC, where:

- The a=OIPF-QoS-Metrics line includes information on the cumulative performance metrics the Service Provider requests from the client for that session (see section 7.2.1, “Performance Monitoring over UNIT-18.”).
- The a=rtcp-xr line includes information on the sample performance metrics the Service Provider requests from the client for that session (see section 9.2.1, “Performance Monitoring over UNIT-18.”)

- Finally, the b=RR line specifies the reporting bandwidth assigned to the OITF in bits per second, see section 2 of RFC 3556 **[SDP-RTCP]**. If this line is not retrieved via SIP OPTIONS, then the OITF SHALL use the RTP default of 5% of the stream bandwidth for RTCP reports.

The use of RTSP SHALL comply with RFC 2326 **[RTSP]** with modifications defined by this specification.

Annex B.2.4 gives guidelines for specifying cumulative metrics to be conveyed using the RTSP Headers defined herein. Similarly to the RTCP case, "OIPF-BasicPerfMonCumulSubset1" is used in this specification as a placeholder and for illustrative purposes.

7.1.1.2.1 Missing SDP parameters Retrieval

When the Cluster controller receives a SIP OPTIONS message to retrieve missing parameters, it SHALL send an RTSP DESCRIBE message to an appropriate CDF. The "Accept" header SHALL be set to "application/sdp".

The CDF SHALL reply with a RTSP 200 OK with the Content-type header set to "application/sdp".

7.1.1.2.2 RTSP Session Setup

The OITF SHALL NOT use the RTSP SETUP message. In the managed network case, the RTSP session setup is initiated with a SIP INVITE.

When receiving a COD session initiation SIP request from the CDN Controller, the Cluster Controller SHALL choose the appropriate CDF and issue an RTSP SETUP message with the following parameters:

- The RTSP URI SHALL have a path that is compatible with the requested content indicated in the user part of the "To:" header of the SIP message.
- The Transport header SHALL contain a "Destination" sub header indicating the IP address o the OITF'.

The CDF SHALL reply with an RTSP 200 OK message to the Cluster Controller.

- The message SHALL contain an RTSP session ID.
- The CSeq SHALL be set to the same value as in the RTSP SETUP request

If the Cluster Controller receives an error message from the CDF, it MAY try another CDF. It MAY also reply with the appropriate SIP error message to the CDN Controller (see section 6.2.2, Content on Demand)

The Cluster Controller SHALL issue a SETUP request for each media line (if the content is FEC protected).

7.1.1.2.3 RTSP Control for media delivery

Handling of Media Control for Starting Playback

On receiving a request from the user to start playback, the OITF SHALL send an RTSP PLAY message to the Cluster Controller using the h-uri attribute received in the SDP (of the SIP 200 OK received as a response to the SIP INVITE request as described in section 6.2.2.1, "Retrieving missing parameters in the SDP prior to session setup using SIP OPTIONS."). If a fully qualified domain name (FQDN) is used in the RTSP URI, the OITF SHALL NOT perform a DNS lookup. For the RTSP PLAY message, the IP address in the IP header SHALL be set to the destination IP address taken from the connection line ("c=" from the m-line of the RTSP stream) in the SDP answer and the port in the TCP header SHALL be taken from the media line ("`m=`") in the SDP answer.

NOTE: The OITF does not perform DNS lookup in order to avoid misaligning the information conveyed in the SDP.

The RTSP fields in the RTSP PLAY message SHALL be filled as follows:

- The RTSP URI SHALL be set to the value retrieved from the SDP h-uri attribute in the case of an absolute URI. If the value of h-URI is a relative URI that is in the form of a media path, then the RTSP absolute URI SHALL be constructed by the OITF using the SDP IP Address (from c-line) and port (from m-line) as the base followed by h-uri value for the media path.
  (ex. `rtsp://10.5.1.72:22554/TV3/823527`)

On receiving a request from the user to modify the playback, the OITF SHALL send an RTSP PLAY message with a request to modify the position, speed and/or direction of playback. The OITF indicates the direction and/or speed of playback by including a `Scale` header or changes the position of playback by including a `Range` header.

- The scale header SHALL be set as follows:
  - 1 indicates normal play;
  - If not 1, the value corresponds to the rate with respect to normal viewing rate;
  - A negative value indicates reverse direction.

If the request is to pause the playback, the OITF SHALL send an RTSP PAUSE message.

On receipt of a RTSP PLAY or PAUSE request, the Cluster Controller SHALL forward the message to the chosen CDF.

The CDF SHALL respond with a 200 OK message. The contents of the 200 OK response SHALL be as follows:

- CSeq SHALL be set to the same value as that in the request.

Handling of Media Control for Retrieving Playback Information

On receiving a request from the user to retrieve playback information, the OITF MAY send an RTSP GET_PARAMETER message. The OITF MAY retrieve the following information:

- position
  The position in the media in seconds.
- scales
  The allowed scales that can be used in the PLAY request. Syntax SHALL be a comma separated array of allowed scales.
- duration
  The total duration in seconds of the media to be played.

Any other parameter not supported by the Cluster Controller used in the GET_PARAMETER request SHALL be rejected by the Cluster Controller with an appropriate error code. An empty body SHALL be allowed for the RTSP keep alive message.

On receipt of the RTSP GET_PARAMETER request, the Cluster Controller SHALL forward the message to the chosen CDF.

The CDF SHALL respond with a 200 OK message with the requested values. The message SHALL be forwarded to the OITF.

Handling of Beginning and End of Stream

On receipt of the beginning-of-stream or end-of-stream indication from the CDF, the Cluster Controller MAY send an RTSP ANNOUNCE to the OITF with an indication that the beginning-of-stream or end-of-stream has been reached. The “Notice” header SHALL be included with the notice code value set to “2104 Start-of-Stream-Reached” or “2101 End-of-Stream Reached”.

Note: The header and code is based on **[RTSP2-AN]**. Note that the RTSP version used in this specification is “1.0” and not “2.0” as in the examples in **[RTSP2-AN]**.

On receipt of the RTSP ANNOUNCE with a beginning of stream or an end-of-stream indication, the OITF MAY take relevant actions to handle the end of stream event (e.g. terminating the session, rewinding the media stream, etc.). The OITF SHALL respond with a RTSP 200 OK.

7.1.1.2.4 RTSP Session Teardown

The OITF SHALL NOT use the RTSP TEARDOWN message.

Note: In the managed network case, the RTSP session teardown is initiated via SIP.

When the Cluster Controller receives a SIP BYE message to teardown a SIP CoD session, the Cluster Controller SHALL send a RTSP TEARDOWN message to the CDF. The CDF SHALL reply with a 200 OK.

The Cluster Controller SHALL issue a TEARDOWN request for each media line (if the content is FEC protected).

7.1.1.2.5 RTSP Messages supported

The OITF acting as an RTSP Client SHALL support RTSP PLAY, RTSP PAUSE, , RTSP GET_PARAMETER, RTSP ANNOUNCE, and RTSP OPTIONs.

The Cluster Controller acting as an RTSP Proxy and RTSP Client SHALL support at least the following messages: RTSP SETUP, RTSP TEARDOWN, RTSP DESCRIBE, RTSP PLAY, RTSP PAUSE, RTSP OPTIONS, and RTSP GET_PARAMETER.

The CDF acting as an RTSP server SHALL support at least the following messages: RTSP SETUP, RTSP TEARDOWN, RTSP DESCRIBE, RTSP PLAY, RTSP PAUSE, RTSP OPTIONS, and RTSP GET_.

7.2 Protocols for Service Access and Control

7.2.1 Performance Monitoring over UNIT-18

QoS metrics can be classified as those that need to be reported regularly, i.e. 'sample metrics', and those that are typically required when the service ends, i.e., 'cumulative metrics'. Periodic RTCP reports are more appropriate for transport of sample metrics (see section. 9.2.1, "Performance Monitoring over UNIT-18"), while on-demand or scheduled RTSP reports are especially suitable for transport of cumulative metrics. In general, an IPTV service might need a combination of both.

This section specifies the OPTIONAL RTSP mechanisms for performance monitoring of CoD services.

If the OITF supports QoS metrics and has been suitably configured to use them, then the CoD session initiation request over HNI-IGI interface SHALL include the selected (i.e. accepted by client) or modified (for re-negotiation) QoS metrics for either the session level or media level.

The *QoS metrics* negotiation SHALL start at the Cluster Controller (CC) on reception of a response to an RTSP DESCRIBE including metrics information embedded in the session description. The RTSP DESCRIBE at the CC is triggered by a SIP OPTIONS request for missing parameters from the CDN Controller, as per section 7.1.1.2.2, "RTSP Session Setup."

On receiving this SETUP request, the Content Delivery Function (CDF) SHALL return the RTSP 200 OK response with the "accepted" QoS metrics (i.e. metrics and reporting values which are identical to the ones in the client's request and accepted by the CDF) and the "re-negotiation" QoS metrics (i.e. metrics and reporting values which are not identical to the ones in the client's request and modified for re-negotiation by the CDF). The echoing of the "accepted" QoS metrics is for re-acknowledging the client's request.

The CDF MAY also reject the changes made by the client, i.e. reject the "re-negotiation" of QoS metrics. If the CDF rejects the changes, it SHALL either set new values and resend the modified metrics back to the client, or it SHALL ignore the "re-negotiation" metrics and not re-acknowledge them. Any QoS metric that has been acknowledged as "accepted" by the CDF SHALL not be re-negotiated, i.e., it SHALL NOT be resent in the "OIPF-QoS-Metrics" header in the next RTSP request and SHALL NOT be re-acknowledged in the next RTSP response.

If the CDF does not approve the modifications made by the client, they SHOULD continue to re-negotiate. However, negotiations SHOULD not exceed 4 round trips, in order to minimize the potential delay of the negotiation process. The negotiation process may delay the start-up of the service and it may be avoided by carefully selecting the value of the *Metrics-Set* parameter in the service information. It must be noted that each time the "QoS-Metrics" header field is sent in an RTSP request, it SHALL also be present in the response corresponding to that particular request. Otherwise, the receiver of the response SHALL assume that the other end does not support QoS metrics.

If there is no DESCRIBE request-response pair sending at the beginning of the RTSP session between the CC and the CDF, it means that the session description is received by other means. If such a description contains the "OIPF-QoS-Metrics" attribute, the negotiation starts at the CC with a SETUP request containing the "OIPF-QoS-Metrics" header.

If the session description does not contain the "OIPF-QoS-Metrics" attribute and the CDF would still like to check whether the client supports the QoS Protocol or not, the CDF SHALL include the "OIPF-QoS-Metrics" header containing the initial QoS metrics in the SETUP response. If the OITF sends the QoS metrics information in the next request (indicating that it supports the QoS Protocol), the negotiation SHALL continue until a mutual agreement is reached or the negotiation limit of 4 round-trips is reached.

To inform the client of the CDF's desire to receive reports for the session, a new SDP attribute is specified to convey the QoS metrics. This attribute is defined inline with section 5.3.3.6 of TS26.234v750 **[PSS]**.

The ABNF definition (See RFC 4234 **[ABNF]**) is as follows:

```
OIPF-QoS-Metrics-Att = “a=” “OIPF-QoS-Metrics” “:”  Measure-Spec *(“,” Measure-Spec) CRLF

    Measure-Spec = ”Metrics “;” Sending-rate [”;” Measure-Range] *([”;” Parameter-Ext])

    Metrics = “cumul-metrics” “=” Metrics-Set / (“{” Metrics-Name *(“|” Metrics-Name) “}”)

    Metrics-Name =  1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except “;”, “,”, “{“ or ”}”

    Metrics-Set = 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except “;”, “,”, “{“ or ”}”

    Sending-Rate = “rate” “=” 1*DIGIT / “End”

    Measure-Range = “range” “:” Ranges-Specifier

    Parameter-Ext = “On”/”Off”/ (1*DIGIT [“.” 1*DIGIT]) / (1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a)
                        / 0x7c / 0x7e))

    Ranges-Specifier = as defined in RFC 2326 [RTSP]

    CRLF = %d13.10
```

This specification defines two new RTSP Headers to negotiate and report the ‘cumulative metrics’ during session setup. These new RTSP headers follow the semantics of [PSS], and are accordingly called *OIPF-QoS-Metrics* and *OIPF-QoS-Feedback*. They SHALL be used as follows:

- *OIPF-QoS-Metrics* SHALL be used for setting up and controlling the reporting of cumulative metrics, i.e. turn on/off reporting, negotiate the set of metrics, its frequency and the report range. This header can be sent in requests and responses of the RTSP Methods SETUP (in unmanaged networks), SET_PARAMETER, OPTIONS (with Session ID) and PLAY. The OITF SHOULD use the OPTIONS (with Session ID) or SET_PARAMETER RTSP methods to turn off the QoS feedback.

  The ABNF **[ABNF]** definition is as follows:

```
QoS-Header = “OIPF-QoS-Metrics” “:” (“Off” / Measure-Spec *(“,” Measure-Spec)) CRLF

    Measure-Spec = Stream-URL ”;” ((Metrics “;” Sending-rate [”;”
                        Measure-Range] *([”;” Parameter-Ext])) /    “Off”)

    Stream-URL = “url” “=”  <”>Rtsp-URL<”>

    Metrics = “cumul-metrics” “=” Metrics-Set / (“{” Metrics-Name *(“|” Metrics-Name) “}”)

    Metrics-Set = 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except “;”, “,”, “{“ or ”}”

    Metrics-Name = 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except “;”, “,”, “{“ or ”}”

    Sending-Rate = “rate” “=” 1*DIGIT / “End”

    Measure-Range = “range” “:” Ranges-Specifier

    Parameter-Ext = “On”/”Off”/ (1*DIGIT [“.” 1*DIGIT]) / (1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) /
                        0x7c / 0x7e))

    Ranges-Specifier = as defined in RFC 2326 [RTSP]

    Rtsp-URL = as defined in RFC 2326 [RTSP]

    CRLF = %d13.10
```

- *OIPF-QoS-Feedback* SHALL be used for sending (or requesting) the actual QoS metrics feedback to (or from) the CDF. This header can be sent in requests and responses of the following RTSP Methods: SET_PARAMETER (or GET_PARAMETER), or PAUSE Methods.

ABNF **[ABNF]** definition follows:

```
Feedback-Header = “OIPF-QoS-Feedback” “:” Feedback-Spec *(“,” Feedback-Spec) CRLF

    Feedback-Spec = Stream-URL “;” 1*(“;” Parameters) [”;” Measure-Range]

    Stream-URL =  “url” “=”  <”>Rtsp-URL<”>

    Metrics-Set = 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except “;”, “,”, “{“ or ”}”

    Parameters = Metrics-Name “=” “{” SP / (Measure *(“|” Measure)) “}”

    Metrics-Name = “1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except “;”, “,”, “{“
                        or ”}”

    Rtsp-URL = as defined in RFC 2326 [RTSP]

    Measure-Range =  “range” “:” Ranges-Specifier

    Ranges-Specifier = as defined in RFC 2326 [RTSP]

    Measure = Value [SP Timestamp]

    Value = ([”-“]1*DIGIT [”.” *DIGIT]) /  1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a)  / 0x7e) ;
                        VCHAR except “;”, “,”, “{“ or ”}”

    Timestamp = NPT-Time

    NPT-Time = as defined in RFC 2326 [RTSP]

    CRLF = %d13.10
```

The OITF SHALL use the SET_PARAMETER method for cumulative QoS metrics reporting, using the OIPF-QoS-Feedback Header defined in this specification. However, in some cases, the RTSP method MAY also be used:

- When the client wants to pause the streaming flow, the QoS information SHOULD be embedded into a PAUSE method. The OITF should not send any QoS reports to the CDF when the media stream is paused, as no media is being exchanged.

The reporting CDF SHALL use the GET_PARAMETER method for retrieving cumulative QoS metrics from the OITF on-demand, using the OIPF-QoS-Feedback Header defined in this specification.

The RTSP header and attribute definitions above are almost identical to those in **[PSS]**. The semantics shall comply with section 5.3.2.3 except for the differences in naming and the following Open IPTV Forum specific changes:

- There is a possibility to simplify the request and reporting of a (potentially large) set of metrics by introducing a *Metrics-Set* attribute in the *OIPF-QoS-Metrics* and *OIPF-QoS-Feedback* headers. i.e., the *Metrics-Set* attribute MAY be used instead of listing each metric explicitly, as these would have empty values during the metrics negotiation phase anyway; this makes messages more compact when the number is large. See the examples in Annex B.2.3

Note the commonalities between the definition of the RTSP Header OIPF-QoS-Metrics and the SDP attribute above. Note also, that the *Stream-URL* is not present in the SDP attribute; this is because the value of the *Stream-URL* is present in the session description and is thus known by the CC at the time of issuing the RTSP SETUP for each media line.

Example message flows are provided in Annex B.2.3. Examples of metrics defintions are given in Annex B.2.4.

8 IGMP and Multicast Protocol

This section defines the protocol for the use of IGMP and Multicast over the following reference points:

- UNIS-13
- UNIS-7
- UNIS-15
- UINS-RMS
- UNIS-6
- UNIS-12

8.1 Protocols for IPTV Service Functions

8.1.1 Scheduled Content

The OITF SHALL support IGMPv3 as described in RFC 3376 (**[IGMP3]**). If the Transport Processing Function supports a lower IGMP version, the backward compatibility rules between the OITF and the Transport Processing Function SHALL conform to **[IGMP3]** section 7.

8.1.1.1 Procedure for Scheduled Content on UNIS-13 with Session Initiation

The use of IGMP on UNIS-13 with session initiation SHALL be as defined in **[TS183063]** sections 8.1.2 and 8.2, where the OITF replaces the UE, the Transport Processing Function replaces the Transport Functions, the BTF replaces the ECF/EFF and the Service Discovery FE/IPTV Metadata Control FE replaces the SSF.

8.1.1.2 Procedure for Scheduled Content on UNIS-13 without Session Initiation

The use of IGMP on UNIS-13 without session initiation SHALL be as defined in section 8.1.1.1, “Procedure for Scheduled Content on UNIS-13”.

In the case there is no session initiation, the procedures described in section 8.1.2 of **[TS183063]** to set the Group/Multicast Address Records will not retrieve any channel or source address information from the session initiation step.

8.2 Protocols for Service Access and Control Function

8.2.1 Service Discovery and Content Selection

8.2.1.1 Protocol on UNIS-15 and UNIS-7

DVB SD&S Transport Protocol (DVBSTP) specified in section 5.4.1 of **[TS102034]** SHALL be used to transport the Service Discovery and Content Metadata related information over multicast.

8.2.1.2 Protocol on UNIS-13

The use of IGMP on UNIS-13 for Service Discovery and Content Selection SHALL be as defined in **[TS183063]** section 8.1.1 where the OITF replaces the UE, the Service Provider Discovery FE replaces the SDF and the Service Discovery FE/Metadata control FE replaces SSF.

8.2.2 Remote Management

8.2.2.1 Protocol on UNI-RMS

The UNI-RMS provides the functions for the remote management of the ITF devices. This interface SHALL be based on DSL Forum TR-069 Remote Management Framework **[TR069]**.

8.3 Protocols for System Infrastructure Functions

8.3.1 Interactive application delivery

8.3.1.1 Protocol over UNIS-6 and UNIS-12

The use of multicast is an OPTIONAL feature for reducing network and server traffic when DAE and PAE applications are delivered to the OITF and AG respectively.

FLUTE **[RFC3926]** SHALL be used when DAE and PAE applications are delivered through multicast. The FDT-Instance XML Schema defined in **[RFC3926]** is extended with two additional attributes: Tags and Priority. The Tags attribute contains a list of tags that the content is associated with. The optional Priority attribute is used by the OITF to determine which content items can be discarded when there is a need to recover memory. The Priority attribute takes values between 1 and 10, with 10 being the highest priority.

The detailed schema extensions are described in Annex M.

The Content-Location element in the FDT-Instance SHALL be the same URI which is used to fetch the object with unicast, e.g., http://server/DAE_pictures/background.jpg. If the OITF has stored an object, it SHALL NOT download the object again using unicast unless it has expired.

If certain parts of the file are lost, the OITF MAY fetch the missing parts using HTTP with range headers.

9 RTP/RTCP

This section defines the protocol for the use of RTP and RTCP over the following reference points:

- UNIT-17
- UNIT-18

9.1 Protocols for IPTV Service Functions

9.1.1 Scheduled Content

Scheduled content is delivered either as a multicast or a unicast stream over UNIT-17.

9.1.1.1 Protocol over UNIT-17

The use of RTP on this reference point SHALL comply with **[TS102034]** section 7.1 and subsection 7.1.1.

9.1.2 CoD

Streamed Content on Demand is delivered as a unicast stream over UNIT-17. RTP and HTTP may be used. This section specifies the use of RTP.

9.1.2.1 Protocol over UNIT-17

RTP SHALL be used on this reference point and SHALL comply with **[TS102034]** section 7.1 and subsection 7.1.1.

The use of RTCP SHALL be compliant to section 9.2.1.

9.2 Service Access and Control

9.2.1 Performance Monitoring over UNIT-18

This section specifies the OPTIONAL setup and reporting of sample metrics for CoD services.

The setup is initiated when the SDP session parameters are retrieved. A CDF requesting sample metrics reports via RTCP SHALL include the performance reporting information in the retrieved SDP information using the a=rtcp-xr SDP attribute as defined below.

RTCP SHALL be used as per subsection 7.1.1.1 of **[TS102034]** with the following additions:

- RTCP Receiver Reports defined by RFC 3550 **[RTP]** SHALL be used to include RTCP XR extended reports following the rules of RFC 3611 **[RTCP-XR]** and as defined further below.
- The RTP default value for RTCP bandwidth of 5% MAY be overridden by using the SDP bandwidth modifiers as specified in RFC 3556 **[SDP-RTCP]**. See section 7.1.1.2 for details.

Unlike the case of cumulative metrics, the sample metrics are not present in the *OIPF-QoS-Metrics* RTSP Header as there is already a framework for reporting metrics using RTCP, namely the RTCP XR extended report as per RFC 3611 **[RTCP-XR]**. A consequence of this is that the sample metrics cannot be negotiated, as RTSP cannot negotiate parameters present in the SDP.

The SDP attributes to define the required sample metrics in accordance with section 5.1 of RFC 3611 **[RTCP-XR]** are as follows:

```
rtcp-xr-attrib = “a=” “rtcp-xr” “:” [xr-format *(SP xr-format)] CRLF

    xr-format = default-OIPF-xr-report / new-OIPF-xr-report

    default-OIPF-xr-report = “OIPF-BasicPerfMonSampleSubset1”

    new-OIPF-xr-report = non-ws-string   ; non-white-space string

    non-ws-string = 1*(%x21-FF)

    CRLF = %d13.10
```

An OITF using RTCP reporting shall support the extended report blocks defined in future versions of this specification. Annex B.2.4 gives guidelines for specifying RTCP XR Extended Report containing sample metrics. *OIPF-BasicPerfMonSampleSubset1* is used in this specification as a formal placeholder and for illustrative purposes.

The OITF may support other RTCP XR extended reports as defined in RFC 3611 **[RTCP-XR]** (e.g., pkt-loss-rle, pkt-dup-rle, pkt-rcpt-times, etc...) but these are not required for the performance monitoring to work correctly.

9.3 Application Layer Forward Error Correction

This section specifies the protocol for Application Layer FEC (AL-FEC) protection of streaming media for Scheduled Content services carried over RTP transport.

Application Layer FEC SHALL conform to [TS102 034] annex E. Only the base layer of DVB-IPTV AL-FEC SHALL be supported. Support of the AL-FEC enhancement layer is out of scope.

DVB AL FEC base layer is signalled in DVB SD&S, as defined in [TS102034] section 5.2.6.2

10 UPnP

10.1 Protocols for System Infrastructure Functions

10.1.1 UPnP Discovery

OITFs that do not support native HNI-IGI do not support UPnP-based IG discovery.

10.1.1.1 Procedure for IG Discovery

10.1.1.1.1 Discovery Sequence

When an OITF powers up, the OITF SHALL automatically discover the IG using the UPnP Discovery Mechanism defined by UPnP Device Architecture **[UPNP]**. A summary of the steps are as follows:

Step 1: An OITF sends the UPnP search request with the search target (urn:oipf-org:device:ig:1) to the specific multicast IP/Port address (239.255.255.250:1900)

Step 2: When an IG receives the search request, the IG sends the response message with its UPnP device description location (URL, e.g. http://IGAddress/Description.xml) to the requester's IP address by HTTP-U protocol

Step 3: The OITF sends the HTTP GET request for retrieving the UPnP device description from the location (e.g. http://IGAddress/Description.xml)

Step 4: The IG sends the response with its UPnP device description, which holds the IG description.

10.1.1.1.2 urn:oipf-org:device:ig:1 device definitions

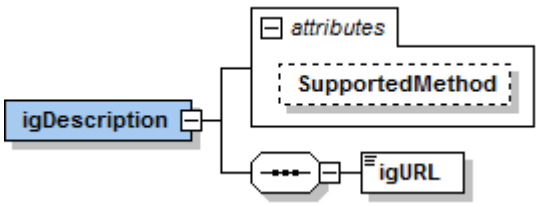
This section defines the urn:oipf-org:device:ig:1 deviceType.

deviceType	Root	R/O	ServiceType	R/O	ServiceID
urn:oipf-org:device:ig:1	Root or Embedded	Required	see below	n/a	n/a

As described above, the urn:oipf-org:device:ig:1 deviceType does not have any specific definition for the serviceType it supports. It may have services of any serviceType.

10.1.1.1.3 IG Description

To interact with the IG, the OITF MUST know the IG URI and possibly a set of supported methods. The device element of the device description document for the urn:oipf-org:device:ig:1 deviceType SHALL contain this information, IG description, which is described as an XML fragment. The XML schema for the IG description is as follows:




```
<schema targetNamespace="urn:oipf-org:device:ig:1"
  xmlns:tns="urn:oipf-org:device:ig:1"
  xmlns="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified">

  <element name="igDescription">
    <complexType>
      <sequence>
```



```
      <element name="igURL" type="anyURI" />
    </sequence>
    <attribute name="SupportedMethod"
      type="tns:Hexadecimal16bit" use="optional" />
  </complexType>
 </element>
 <simpleType name="Hexadecimal16bit">
  <restriction base="string">
    <pattern value="[0-9a-fA-F]{1,4}" />
  </restriction>
 </simpleType>
</schema>
```

Element/Attribute Name	Description
igDescription	The root element of the IG Description XML fragment
@SupportedMethod	Hexadecimal16bit to describe the methods supported by the IG over the HNI-IGI interface (section 5.5.1, “OITF-IG Interface (HNI-IGI)”), which correspondcorrespond to SIP methods and other functionality. Each bit represents a supported method as follows : 0001 : REGISTER 0002 : INVITE 0004 : BYE 0008 : CANCEL 0010 : SUBSCRIBE 0020 : NOTIFY 0040 : PUBLISH 0080 : MESSAGE 0100 : OPTIONS 0200 : GBA Authentication (0400~8000 : Upper 6 bits are currently not assigned but these bits could be used for other methods defined later) The value of this attribute is the result of a 16 bit OR operation with representative values of supported methods. If the SupportedMethod attribute is not described, the IG shall support all methods.
igURL	IG URL for the HNI-IGI (refer to section 5.5.1, “OITF-IG Interface (HNI-IGI).”) It MAY be relative to base URL (URLBase of uPnP device description if it exists or the URL from which the device description was retrieved). It SHALL NOT exceed 256 bytes in URI-escapes UTF-8 encoded form.

The namespace of this fragment is “urn:oipf-org:device:ig:1”.

Example:

```
<ig:igDescription xmlns:ig="urn:oipf-org:device:ig:1" SupportedMethod="01ff">
<ig:igURL>
```



```
  http://192.168.0.2/IG/
</ig:igURL>
</ig:igDescription>
```

10.1.1.2 Procedure for AG Discovery

10.1.1.2.1 Discovery Sequence

When an OITF powers up, the OITF SHALL automatically discover the AG using the UPnP Discovery Mechanism defined by UPnP Device Architecture **[UPNP]**. A summary of the steps are as follows:

Step 1: The OITF sends the UPnP search request with the search target (urn:oipf-org:device:ag:1) to the specific multicast IP/Port address (239.255.255.250:1900)

Step 2: When an AG receives the search request, the AG sends the response message with its UPnP device description location (URL, e.g. http://AGAddress/Description.xml) to the requester's IP address by HTTP-U protocol

Step 3: The OITF sends the HTTP GET request for retrieving the UPnP device description from the location (e.g. http://AGAddress/Description.xml)

Step 4: The AG sends the response with its UPnP device description, which holds the AG description.

10.1.1.2.2 urn:oipf-org:device:ag:1 device definitions

This section defines the urn:oipf-org:device:ag:1 deviceType.

deviceType	Root	R/O	ServiceType	R/O	ServiceID
urn:oipf-org:device:ag:1	Root or Embedded	Required	see below	n/a	n/a

As described above, the urn:oipf-org:device:ag:1 deviceType does not have any specific definition for serviceType it supports. It may have services of any serviceType.

10.1.1.2.3 AG Description

The OITF may interact with the AG in one of two ways.

1. The applications running in the AG MAY provide remote UI as defined by the XML UI listing in CEA-2014 **[CEA2014A]**.

2. The applications running in the AG MAY provide non-UI based services, for example listening for XML HTTP Requests from DAE applications. No specific methods or services are defined.

To interact with the AG, the OITF MUST know the AG URL and possibly a set of supported methods. The device element of the device description document for the urn:oipf-org:device:ag:1 deviceType SHALL contain this information, AG description, which is described as an XML fragment. The XML schema for the AG description is as follows:

```
<schema targetNamespace="urn:oipf-org:device:ag:1"
  xmlns:tns="urn:oipf-org:device:ag:1"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <element name="agDescription">
    <complexType>
      <sequence>
        <element name="agDefaultURL" type="anyURI" />
        <element name="agUIServerURL" type="anyURI" minOccurs="0" />
      </sequence>
    </complexType>
  </element>
</schema>
```


Element/Attribute Name	Description
agDescription	The root element of the AG Description XML fragment
agDefaultURL	URL describing default address of AG, e.g. http://10.1.1.2
agUIServerURL	URL or URLs for remote UI provided by applications running on the AG

The namespace of this fragment is “urn:oipf-org:device:ag:1”.

10.1.1.3 Procedure for CSPG-DTCP Discovery

10.1.1.3.1 Discovery Sequence

When an OITF powers up, the OITF SHALL automatically discover the CSPG-DTCP using the UPnP Discovery Mechanism defined by UPnP Device Architecture **[UPNP]**. A summary of the steps are as follows

Step 1: An OITF sends the UPnP search request with the search target (urn:oipf-org:device:cspg-dtcp:1) to the specific multicast IP/Port address (239.255.255.250:1900).

Step 2: When a CGPG-DTCP receives the search request, the CSPG-DTCP sends the response message with its UPnP device description location (URL, e.g. http://CSPGDTCPAddress/Description.xml) to the requester’s IP address by HTTP-U protocol.

Step 3: The OITF sends the HTTP GET request for retrieving UPnP device description with the location (e.g. http://CSPGDTCPAddress/Description.xml).

Step 4: The CSPG-DTCP sends the response with its UPnP device description which holds the CSPG-DTCP description.

10.1.1.3.2 urn:oipf-org:device:cspg-dtcp:1 device definitions

This section defines the urn:oipf-org:device:cspg-dtcp:1 deviceType.

deviceType	Root	R/O	ServiceType	R/O	ServiceID
urn:oipf-org:device:cspg-dtcp:1	Root or Embedded	Required	see below	n/a	n/a

As described above, the urn:oipf-org:device:cspg-dtcp:1 deviceType does not have any specific definition for serviceType it has. It may have services of any serviceType.

10.1.1.3.3 CSPG-DTCP Description

To interact with the CSPG-DTCP, the OITF MUST know which DRM system is supported, the port number used for DTCP key exchange and on which port the different content access protocols are proxied. The device element of the urn:oipf-org:device:cspg-dtcp:1 deviceType SHALL contain this information, and is described based on the following XML schema:

```
<schema targetNamespace="urn:oipf-org:device:cspg-dtcp:1"
  xmlns:tns="urn:oipf:device:cspg-dtcp:1"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <element name="cspgdtcpDescription">
    <complexType>
      <sequence>
        <element name="DtcpPort" type="integer" />
        <element name="HttpProxyPort" type="integer" />
```



```
      <element name="RtspProxyPort" type="integer" />
      <element name="DRMSystemID" type="anyURI" maxOccurs="unbounded" />
    </sequence>
  </complexType>
 </element>
</schema>
```

Element/Attribute Name	Description
cspgdtcpDescription	The root element of CSPG-DTCP Description XML fragment
DtcpPort	TCP port number for DTCP-AKE
HttpProxyPort	TCP port number for HTTP proxy in CSPG-DTCP
RtspProxyPort	TCP port number for RTSP proxy in CSPG-DTCP
DRMSystemID	Supported DRM system. The format of this attribute complies with DRMSystemId as defined in DRMControlInformation extension in **[META]**.

The namespace of this fragment is “urn:oipf-org:device:cspg-dtcp:1”.

Example:

```
<cg:cspgdtcpDescription xmlns:cg="urn:oipf-org:device:cspg-dtcp:1">
  <cg:DtcpPort>12345</cg:DtcpPort>
  <cg:HttpProxyPort>12346</cg:HttpProxyPort>
  <cg:RtspProxyPort>12347</cg:RtspProxyPort>
  <cg:DRMSystemID>urn:dvb:casystemid:12348</cg:DRMSystemID>
  <cg:DRMSystemID>urn:dvb:casystemid:12349</cg:DRMSystemID>
</cg:cspgdtcpDescription>
```


11 DLNA

11.1 DLNA Function

DLNA Function is an OPTIONAL gateway function which serves IPTV content to other DLNA devices in a consumer network. It converts the IPTV protocols, such as metadata access and media delivery protocols, to DLNA protocols. It MAY also convert media format and content protection schemes, if necessary. The interface between the DLNA Function of the OITF and DLNA devices SHALL be compliant with the DLNA Networked Device Interoperability Guidelines (October 2006) **[DLNA]**.

12 DHCP

This section defines the protocol for the use of DHCP over the following reference points:

- UNIT-16

12.1 Protocols for System Infrastructure Functions

12.1.1 Network Attachment

Network attachment provides IP addresses and configuration information to elements in the Consumer Domain prior to any other action regarding IPTV services. The provision and management of IP addresses has two main aspects.

IP address management within the Consumer Network: Deals with the attachment of the IG, AG and OITF to the WAN Gateway. The WAN Gateway SHALL act as a DHCP Server and a NAT. This type of attachment allows the IG, AG and OITF to communicate with each other within the consumer network. In the unmanaged network model, this allows the OITF to send and receive messages to and from the Internet.

IP address management for communication with the Provider Network (Managed Network model only): 2 cases are supported.

- MANDATORY: The WAN Gateway translates the in-home IP address to an IP address recognizable to the provider's addressing plan. In this case a NAT SHALL be supported.
- OPTIONAL: The WAN Gateway assigns an IP address to the IG, AG and OITF from the managed network's IP addressing pool. In this case, NAT is not required.

Configuration information (e.g. DNS server) SHALL be provided to the OITF, AG and IG

At power up all devices in the consumer network (OITF, IG, AG and other devices) SHALL request an IP address and network configuration parameters from the WAN gateway using DHCP.

12.1.1.1 DHCP Option Usage

12.1.1.1.1 Common Options

The following is the minimum set of DHCP options defined in RFC 3442 **[CLSLESS]** and RFC 2132 **[DHCP-OPT]** that SHALL be used by the OITF, IG and AG when requesting DHCP configuration information from the WAN Gateway:

- Option 1: Subnet Mask
- Option 6: DNS
- Option 61: Client identifier. In this specification, the DHCP Client Identifier used in OITF devices is the deviceID, defined as follows:.
    - deviceID - Identifies the device. It SHALL be unique within the home network and SHALL NOT change between restarts. The deviceID SHALL be the SHA-1 hash of the MAC address of the interface used to connect to the IPTV service as bytes concatenated with the domain name received via DHCP option 15 in ASCII characters:
        - deviceID=SHA-1(X)
          where: X: = (MAC address as bytes) + (domain name in ASCII characters) and the '+' denoted the concatenation operation. SHA-1 SHALL be used as specified in **[SHA-1]**. The domain name SHALL be set to the domain name received via DHCP option 15 (see section 12.1.1.1.2, "Option 15")

For the IG and AG to support TR-069 based remote management, the following SHALL be supported:

- Option 43: Vendor Specific Information is used to retrieve the Remote Management Server IP address or fully qualified domain name (FQDN). This information SHALL be provided by the DHCP server.
- Option 60: Vendor Class identifier is used to indicate to the DHCP server that it is compliant with the Broadband Forum TR-069 specification. The vendor-class identifier SHALL be set to “IG_IPTV” or “AG_IPTV” by the IG or AG, respectively.

If the OITF supports TR-069 based remote management, the following SHALL be supported:

- Option 43: Vendor Specific Information is used to retrieve the Remote Management Server IP address or fully qualified domain name (FQDN). This information SHALL be provided by the DHCP server.
- Option 60: Vendor Class identifier is used to indicate to the DHCP server that it is compliant with the Broadband Forum TR-069 specification. The vendor-class identifier SHALL be set to “OITF_IPTV” by the OITF.

For Managed Networks, the following SHALL be supported:

- Option 120: The address of the P-CSCF as per section 7.1.1 of ETSI TS 183 019 Network Attachment: User-Network protocol Interface Definitions. [TS183019]

12.1.1.1.2 Option 15

Option 15 SHALL be used by the OITF to request the domain name used to generate the device identity (deviceID) as per section 6.3.2.1, “User Identity Modelling”.

12.1.1.1.3 Option 124/125

The OITF SHALL send a Vendor–Identifying Vendor Class option 124 as specified in RFC 3925 **[DHCP-VND]** when it requests a DHCP lease from the WAN Gateway. The option is specified with an enterprise-number and the vendor-class-data identifier as “OITF_IPTV”.

The DHCP server delivers the Service Provider Discovery entry point via Vendor-Identifying Vendor-Specific Information DHCP option 125 as defined in RFC 3925 **[DHCP-VND]**.

In this specification, the Service Provider Discovery entry points used are either:

- The FQDN/IP address of the IG, as per Annex G.1, “OITF Start up High-Level Procedure”, or
- The FQDN/IP address of the Service Provider Discovery Functional Entity (SP Discovery FE), as per section 6.3.1, “Service Provider discovery”.

Format of DHCP payload

The format of the vendor-specific binary buffer containing addresses returned by the DHCP server is a list of sub-options starting with sub-option number (one byte), sub-option length (one byte) and sub-option value (list of bytes).

The following vendor-specific sub-option is defined:

- Sub-Option: IPTV-ENTRYPOINT: Code=0x01. This option carries either an IP Address or a fully-qualified domain name, as determined by a one byte “enc” field is used to indicate the type of encoding.
    - If the “enc” field has a value of 0x01, then this indicates an IP Address. The “enc” field is followed by 4 bytes corresponding to the IP Address. This value is used for the Service Provider Discovery Entry point function.
    - If the “enc” field has a value of 0x02, then this indicates a FQDN (Fully-Qualified Domain Name). This value is used for the Service Provider Discovery Entry point function.
    - The code of 0xFF is used to indicate end of the buffer

13 UDP

13.1 Protocols for IPTV Service Functions

13.1.1 Scheduled Content

13.1.1.1 Protocol over UNIT-17

When MPEG2-TS are encapsulated directly in UDP (User Datagram Protocol) the encapsulation SHALL conform to ETSI TS 102 034 **[TS102034]** section 7.1.2.

The use of RTP or direct UDP encapsulation SHALL be signalled by Service Discovery and Selection for Multicast. For unicast in the managed model, SIP OPTIONS SHALL be utilized for the signalling.

In addition, it SHOULD be possible for an OITF to detect the usage of RTP or direct UDP encapsulation by looking for the value 0x47 in the first byte after the UDP header. In case of Direct UDP encapsulation this is the first byte of a 188 byte MPEG2-TS packet which always have the value 0x47 (synchronization byte of transport stream header. For any other value then RTP encapsulation is used).

13.1.2 CoD

13.1.2.1 Protocol over UNIT-17

The use of UDP on this reference point SHALL be as specified in section 13.1.1.1, “Protocol over UNIT-17.”

Annex A Void

Annex B Example of IPTV Protocol Sequences (informative)

B.1 IPTV Service Functions Protocol Sequences

B.1.1 COD Sequences

B.1.1.1 RTSP specific usage on UNIS-11 and NPI-10 for the managed model

In this example, the RTSP delivery parameters have been obtained as indicated in section 6.2.2.2, “Procedure for Unicast Service Session Initiation.”

The RTSP URI is: rtsp://Cluster.orangeCDN.net/chevaliers_du_ciel

The session ID is 940211290776250

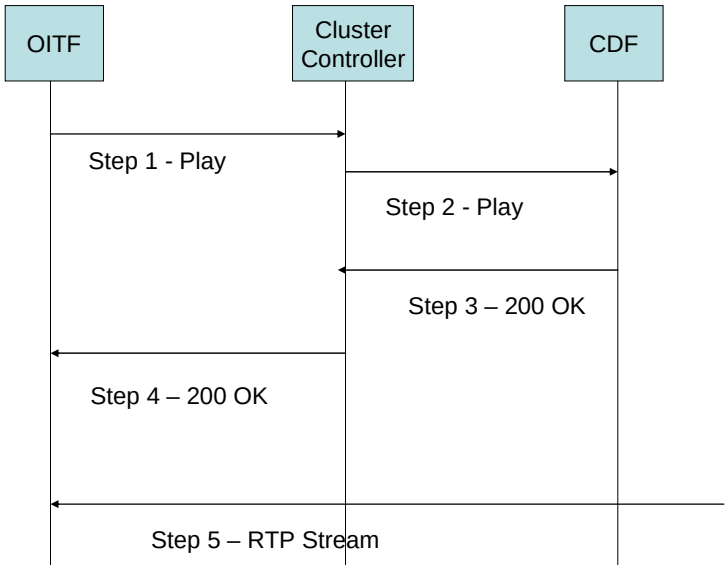


Figure 4: RTSP Procedure on UNIS-11 for managed model

Step 1: The OITF sends an RTSP PLAY to the Cluster Controller

```
PLAY  rtsp://Cluster.orangeCDN.net/chevaliers_du_ciel
    CSeq: 1981
    Session: 940211290776250
```

Step 2: The Cluster Controller forwards the PLAY message to the CDF

```
PLAY  rtsp://server1.Cluster.orangeCDN.net/chevaliers_du_ciel
    CSeq: 1981
    Session: 940211290776250
```

Step 3: The CDF replies to the Cluster Controller


```
200 OK
    CSeq: 1981
    Session: 940211290776250
```

Step 4: The Cluster Controller replies to the OITF with the appropriate RTSP session ID

```
200 OK
    CSeq: 1981
    Session: 940211290776250
```

Step 5: The RTP media starts

B.1.1.2 RTSP specific usage on UNIS-11 and NPI-10 for the unmanaged model

The following example is only one example of performing redirection at initiation using the 303 Moved message. It does not take into account the effects of Network Address Translation (NAT) (See section 7.1.1.1, "RTSP Profile for the unmanaged model over UNIS-11 and NPI-10.")

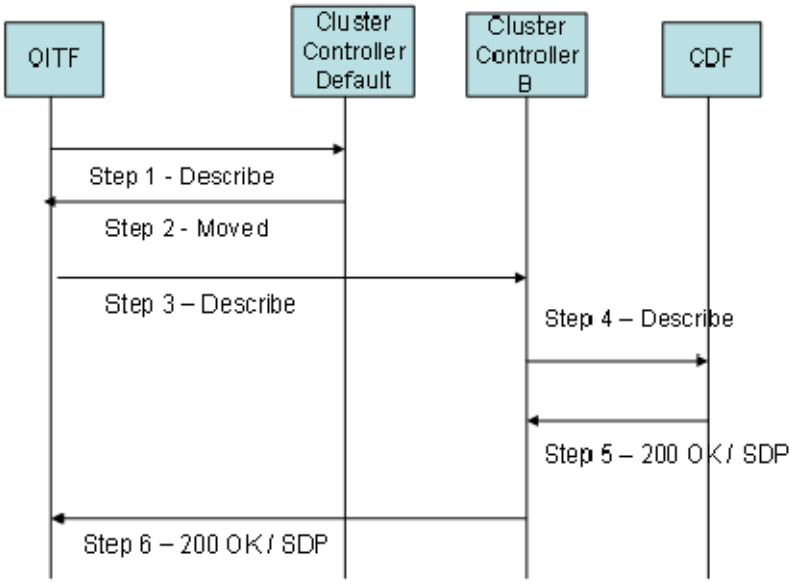


Figure 5: RTSP Usage for COD on UNIS-11 and NPI-10

Step 1: The OITF to the Cluster Controller

```
DESCRIBE rtsp://Cluster.orangeCDN.net/chevaliers_du_ciel RTSP/1.0
    CSeq 1306
    Accept: application/sdp
```

Step 2: The Cluster Controller responds to the OITF indicating redirection to Cluster Controller B

```
RTSP/1.0 302 Moved Temporarily
    CSeq 1306
    Location: rtsp://Cluster_B.orangeCDN.net/ chevaliers_du_ciel  RTSP/1.0
```

Step 3: The OITF sends a DESCRIBE to the indicated Cluster Controller

```
DESCRIBE rtsp://Cluster_B.orangeCDN.net/chevaliers_du_ciel
    CSeq: 1979
    Accept: application/sdp
```

Step 4: The Cluster Controller chooses the appropriate CDF and forwards the DESCRIBE message to it

```
DESCRIBE rtsp://Server1.orangeCDN.net/chevaliers_du_ciel  RTSP/1.0
    Cseq: 1979
    Accept: application/sdp
```

Step 5: The CDF replies to the Cluster Controller with the appropriate SDP

```
200 OK
    Cseq: 1979
    Content-Type: application/sdp
    Content length: …..
    //// SDP////
```

Step 6: The Cluster Controller replies to OITF with the appropriate SDP

```
200 OK
    CSeq: 1979
    Content-Type: application/sdp
    Content length: …
    ////SDP ////
```

B.2 Service Access and Control Function Protocol Sequences

B.2.1 Authentication

B.2.1.1 User Registration and Authentication in a Managed Model

B.2.1.1.1 Default User Identities Registration

The default user IMS Public Identity (IMPU) allocated to the subscription will be automatically registered in the provider network whenever the OITF is turned on.

Figure 6 shows a typical call flow for a default public identity registering in a provider network. The following is a brief description of the steps:

Step 1: The procedure is triggered automatically without any user intervention.

- The OITF issues an HTTP POST request (See section 5.3.6.1, “Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface”).

Step 2: The IG validates that the request (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8”).

Step 3: The IG performs the normal IMS registration procedure (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8”). If the IG does not perform IMS registration because the default identity is already registered (another OITF is activated), the IG still maintains a binding between the IMPU and the new contact (OITF IP address). This means that the IG will

have to fork an incoming SIP request intended for the default IMPU to all OITF entities that are currently active in a consumer network.

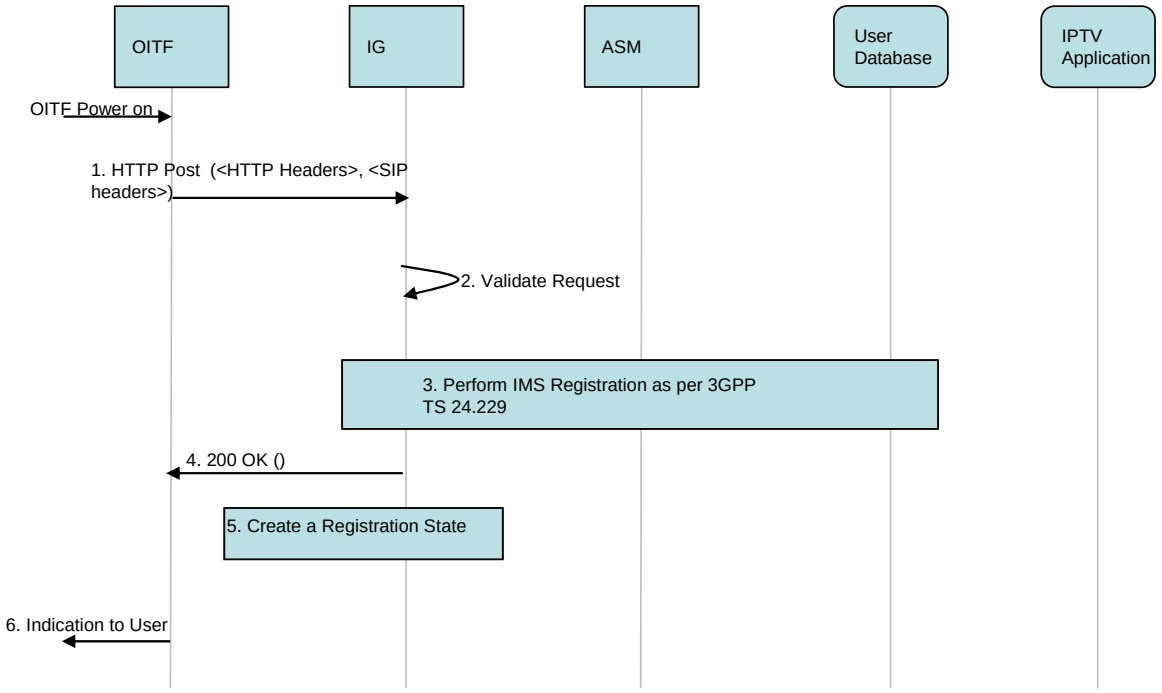


Figure 6: Default IMS Public identity Registration procedure in a managed model

Step 4: The IG returns the outcome of the registration process to the OITF (See section 5.3.6.1, “Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface”).

Step 5: If the result of the registration procedure is successful, a registration state is created and maintained in the IG which is stateful to the registration process until such time as a de-registration occurs. (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8.”

Step 6: An indication is sent to the user that includes the outcome of the registration process. (See section 5.3.6.1, “Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface”).

B.2.1.1.2 IPTV End User Registration

Figure 7 shows a typical call flow for an IPTV end-user registering a specific IMPU in a provider network. The following is a brief description of the steps:

Step 1: The procedure can be triggered by the user wanting to register himself (a specific IMPU associated with him) in the provider network. Other options (e.g. Configuration) can also trigger the procedure.

Step 2: The IG validates the HTTP POST request (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8.”)

Step 3: The IG performs the normal IMS registration procedure (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8”). The IG does not perform an IMS registration if this IMPU is already registered (i.e. another OITF is activated with the same IMPU registered), the IG maintains a binding between the IMPU and the new contact (OITF IP address). This means that the IG will have to fork an incoming SIP request intended for this IMPU to all OITF entities that are currently active in a consumer Network and listed as a contact address for this IMPU.

Step 4: The IG returns the outcome of the registration process to the OITF (See section 5.3.6.1, “Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface”).

Step 5: If the result of the registration procedure is successful, a registration state is created and maintained in the IG which is stateful to the registration process until such time as a de-registration occurs. (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8.”)

Steps 6-7: An indication is sent to the user that includes the outcome of the registration process. (See section 5.3.6.1, "Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface").

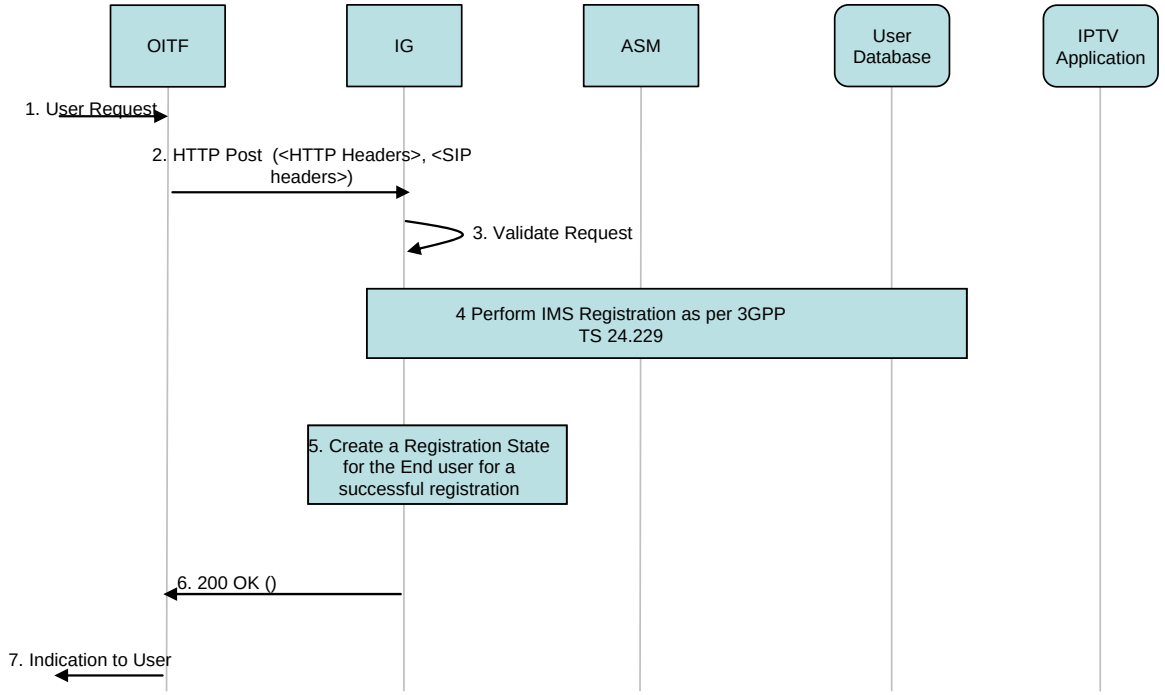


Figure 7: IPTV end-user IMPU Registration procedure in a managed model

B.2.1.1.3 IPTV End User De-registration

User de-registration is similar to user registration (See section 6.3.2.2, "Procedure for User Registration and Authentication in a Managed Model on UNIS-8").

The call flow for the de-registration process is shown in Figure 8.

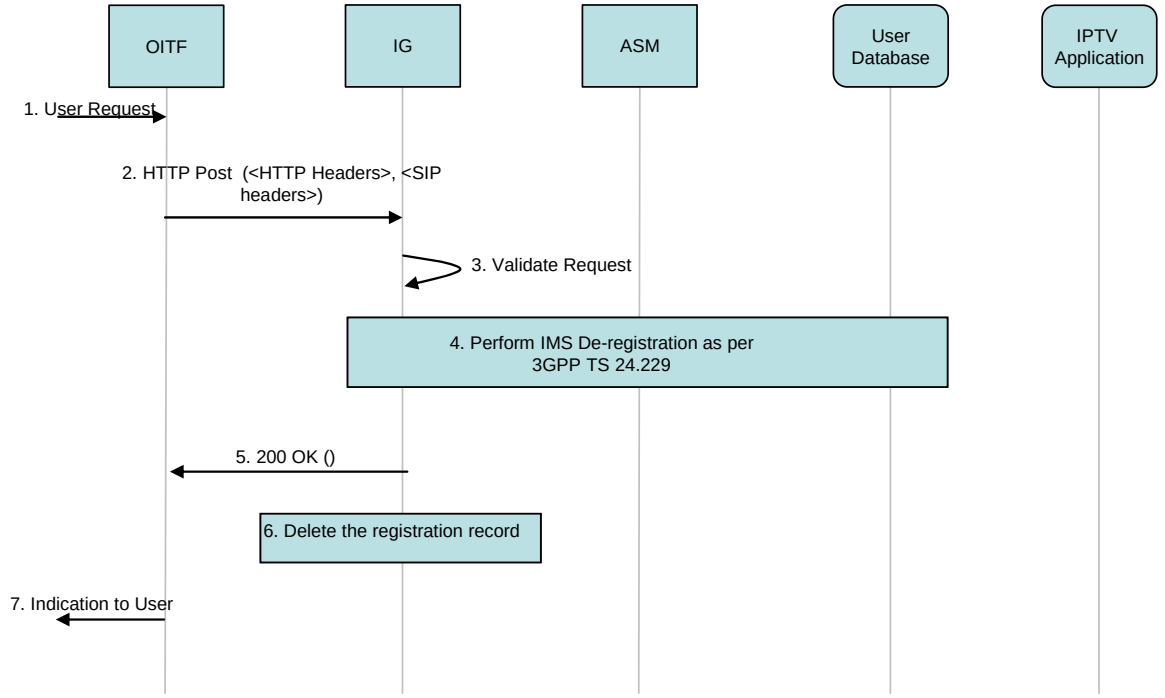


Figure 8: IPTV end-user De-registration procedure in a managed model

B.2.1.1.4 IPTV Default User De-registration

Default public identity de-registration (see section 5.3.6.1.2, “User De-registration” and section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8”).

The call flow for the de-registration process is shown in Figure 9.

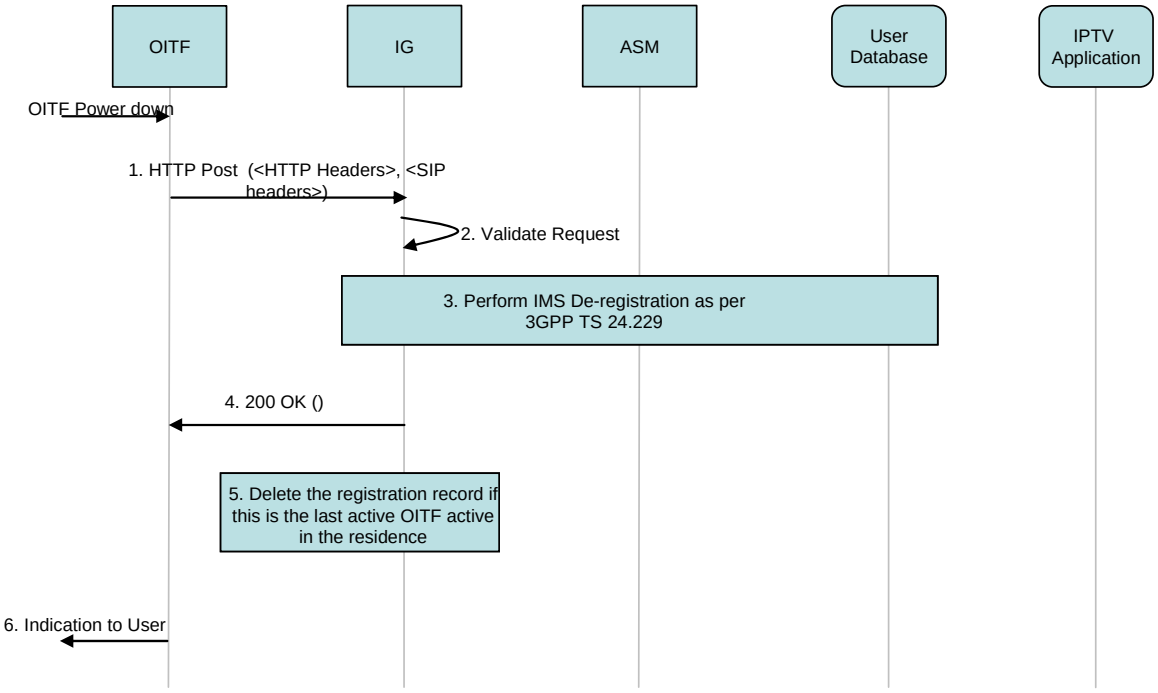


Figure 9: IPTV Default Identity De-registration procedure in a managed model

B.2.1.1.5 Subscription to the registration-state event package

Following the completion of a successful registration, for a default public identity or the IMPU associated with a specific IPTV end-user, the registration application SHALL subscribe to the Registration event. This is mandatory in order to notify the OITF of any event concerning registration (e.g., a registration timeout). This allows the application to take appropriate action, such as re-registering, etc.

Figure 10 shows a typical call flow for subscription to the registration event. The following is a brief description of the steps:

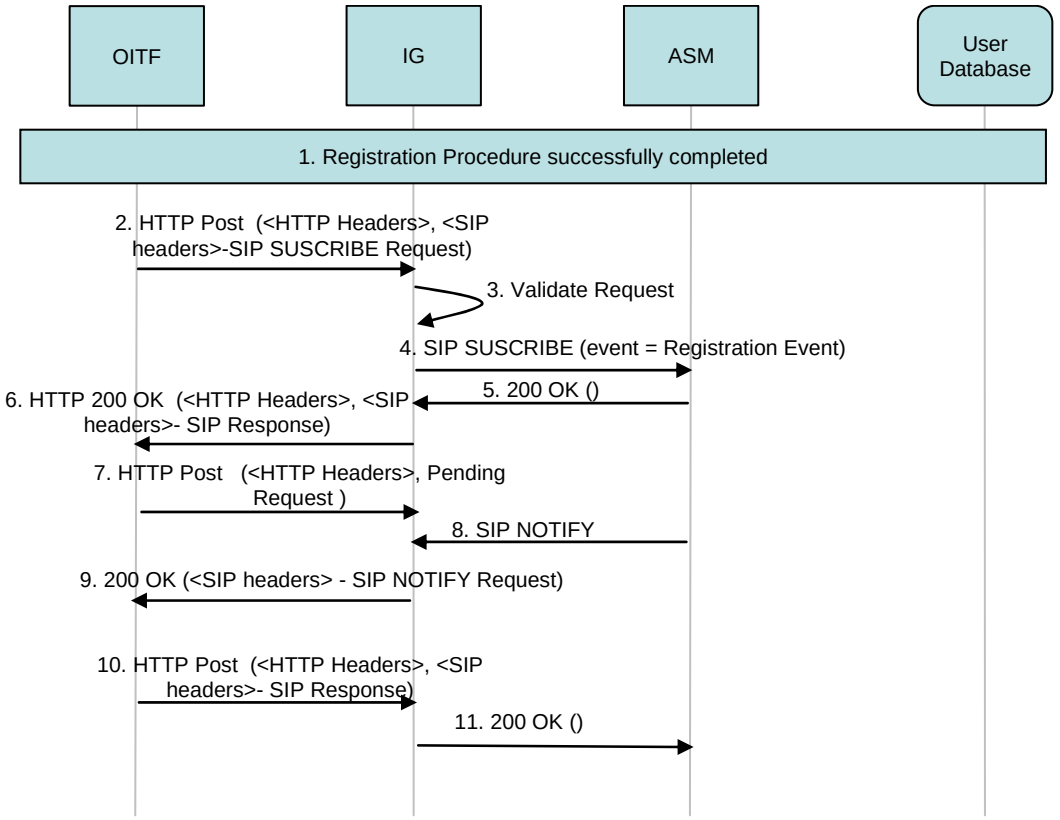


Figure 10: Call flow for subscription to the registration event

Step 1: The OITF concludes a successful registration for a default identity or a specific IMPU associated with an IPTV user.

Step 2: Immediately following a successful registration, the OITF issues an HTTP POST request for subscription to the Registration event. (See section 5.3.6.1.4, “Procedure for Subscription to the Registration Event Package.”)

Step 3: The IG validates that the request. (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8”)

Step 4: The IG performs the normal IMS Subscription process (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8.”)

Step 5: The 200 OK is received from the network. (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8.”)

Step 6: The IG returns an HTTP 200 OK response to the HTTP POST that includes the SIP 200 OK response to the SIP SUBSCRIBE (See section 5.3.6.1.4, “Procedure for Subscription to the Registration Event Package.”)

Steps 7-11: The mechanism for receiving and acknowledging the SIP NOTIFY from the network are covered in Steps 7 to 11. (See section 5.3.6.1.4, “Procedure for Subscription to the Registration Event Package.”)

B.2.2 IPTV Service Profile Manipulation through XCAP

This section shows an example flow for Service Profile Management based on XCAP for IPTV Service Profile access and manipulation.

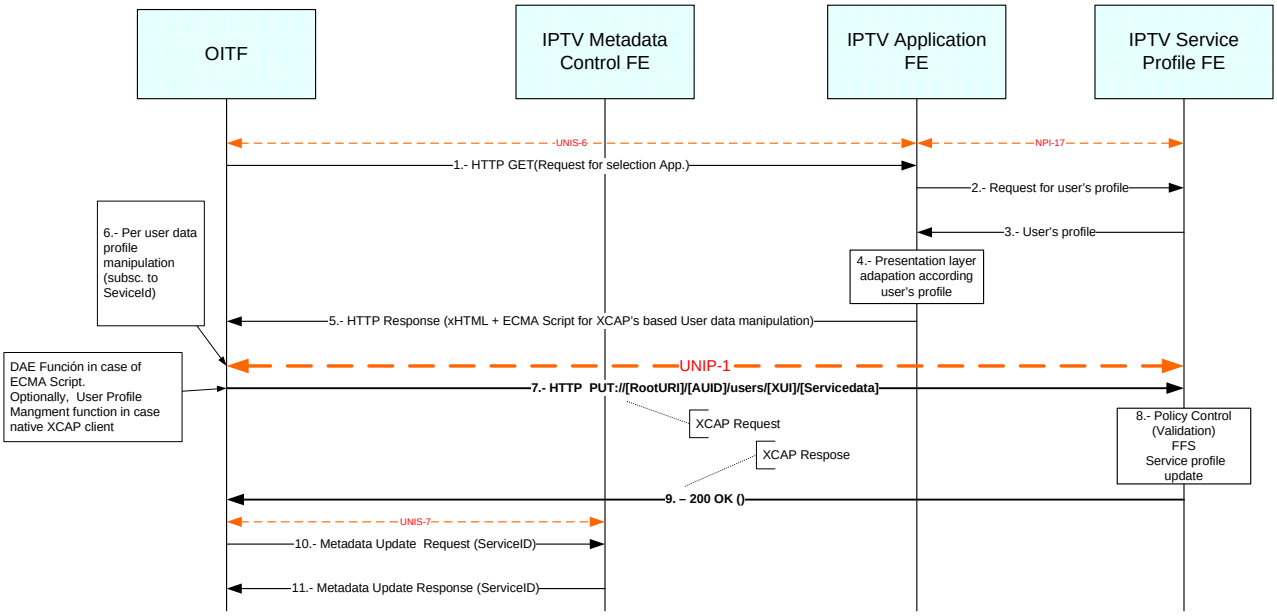


Figure 11: Service Profile Management Based on XCAP

Step 1: After OITF start up, registration and authentication, the DAE client in the OITF automatically requests the service related presentation layer through the UNIS-6 interface. The User is identified via a HTTP header or specific parameters in the URI.

Step 2-3: The IPTV Application FE retrieves the IPTV Service Profile associated with this user from the IPTV Service Profile FE.

Step 4: The IPTV Application FE customises the pages according to the IPTV Service Profile.

Step 5: The personalised presentation pages are downloaded to the OITF. If the IPTV Service Profile explicitly indicates permission for service profile manipulation, an ECMA script supporting XCAP as defined in RFC 4825 **[XCAP]** is also downloaded.

Step 6- 7: When the user decides to add or update a specific service, an XCAP request is sent. The XUI parameter will carry the IMPU of the User.

Note. Optionally, an embedded XCAP client could be supported.

Note that UNIP-1 reference point is used by this ECMA script/DAE application.

Step 8: IPTV Service Profile FE shall validate the request. If validated, the subscription profile data is updated.

Step 9: A HTTP 200 OK is returned as successful answer to the request.

Step 10: *If required*, the Metadata Client in the OITF will request Metadata Control FE ,the metadata related to the service the user has subscribed to. The access is in Pull mode, but does not preclude accessing Metadata in a multicast mode.

Step 11: Carries a response back from the Metadata Control FE via the Metadata client to the DAE Application.

B.2.3 Setup of RTSP/RTCP performance monitoring for CoD Session in Managed Networks over UNIT-18

In this example, it is assume that the OITF has retrieved the SDP information via SIP OPTIONS and the RTSP DESCRIBE between CC and CDF as per section 7.1.1.2.1, “Missing SDP parameters Retrieval”. Specifically, the SDP information consists of two “a=” lines each describing the cumulative and sample reporting metrics the Service Provider wants to receive for the media, and optionally a b=RR: line specifying the receiver’s bandwidth for RTCP reporting. e.g.:

a=OIPF-QoS-Metrics:cumul-metrics=OIPF-BasicPerfMonCumulSubset1;rate=2;
a=rtcp-xr:OIPF-BasicPerfMonSampleSubset1
b=RR:1000

The CoD session is initiated via SIP and the Cluster Controller triggers an RTSP SETUP to the Content Delivery Function (CDF). The request contains the QoS Metrics included in the request by the OITF; these may be exactly the same as the values in the RTSP SDP or the OITF may request to change some values due to reasons such as bandwidth limitations. Assuming the change is acceptable to the CDF, the CDF acknowledges the request by echoing the QoS Metrics Header and its content in the response. The CC forwards the response downstream in the form of a SIP response message.

In the example below, the OITF has agreed to report the cumulative QoS metrics as proposed by the server. This set of metrics is only an example.

Note that by selecting and setting up a particular stream, the OITF is also accepting the request to support sample metrics reporting. This is standard RTSP behaviour.

In the call flow below, only the RTSP messages between CC and CDF are included (not all headers are shown in the examples):

CC→CDF:

SETUP rtsp://example.com/foo/bar/baz.rm RTSP/1.0
CSeq: 1
OIPF-QoS-Metrics: url:"rtsp://example.com/foo/bar/baz.rm";
cumul-metrics=OIPF-BasicPerfMonCumulSubset1;rate=2

CDF→CC:

RTSP/1.0 200 OK
CSeq: 1
OIPF-QoS-Metrics: url:"rtsp://example.com/foo/bar/baz.rm";
cumul-metrics=OIPF-BasicPerfMonCumulSubset1;rate=2

After the time in seconds specified by the “rate=2” parameter, the OITF will send its first report using a SET_PARAMATER request with the OIPF-QoS-Feedback Header and the parameters belonging to the metrics set (in this case reduced for clarity).

OITF→CDF:

SET_PARAMETER rtsp://example.com/foo/bar/baz.rm RTSP/1.0
CSeq: 3
OIPF-QoS-Feedback: url:"rtsp://example.com/foo/bar/baz.rm"; //
PacketsDiscarded={15};PacketsOutOFSequence={2}; //
PacketsRecieved={151}

In order to enable the server to request QoS metrics on-demand, the GET_PARAMETER method is used.

CDF→OITF:

GET_PARAMETER rtsp://example.com/foo/bar/baz.rm RTSP/1.0
CSeq: 15
Session: 13320
OIF-QoS-Feedback: url:"rtsp://example.com/foo/bar/baz.rm"; //
OIF-BasicPerfMonCumulSubset1

OITF→CDF:

```
RTSP/1.0 200 OK
    CSeq: 15
    Session: 13320
    OIF-QoS-Feedback: url:"rtsp://example.com/foo/bar/baz.rm";//
                        PacketsDiscarded={125};PacketsOutOfSequence={20};//
                        PacketsReceived={2651};PacketsLost={7};DecodedFrames={1034}; //
                        LostFrames={2};DecodingErrors={25}
```

B.2.4 Specifying metrics for RTSP/RTCP performance monitoring

This annex provides examples of how to specify Open IPTV Forum specific metrics.

Cumulative metrics are defined in the following manner. For illustrative purposes, the metrics set is named `OIPF-BasicPerfMonCumulSubset1.` The following cumulative metrics values from TR-135 **[TR135]** have been selected :

- from the .STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.RTPStats. object:
  - PacketsDiscarded, or late packets
  - PacketsOutOfSequence, or reordered packets
  - PacketsReceived and,
  - PacketsLost, which is equal to the value of “cumulative number of packets lost” in the RTCP Receiver Report.
- from the .STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.VideoDecoderStats. object:
  - DecodedFrames and,
  - LostFrames
- from the .STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.AudioDecoderStats. object:
  - DecodingErrors

As specified in section 7.2.1, “Performance Monitoring over UNIT-18”, these metrics are set up using the OIPF-QoS-Metrics header and reported using OIPF-QoS-Feedback header, e.g., a CC indicating setup of the `OIPF-BasicPerfMonCumulSubset1` set of metrics would send:

CC→CDF:

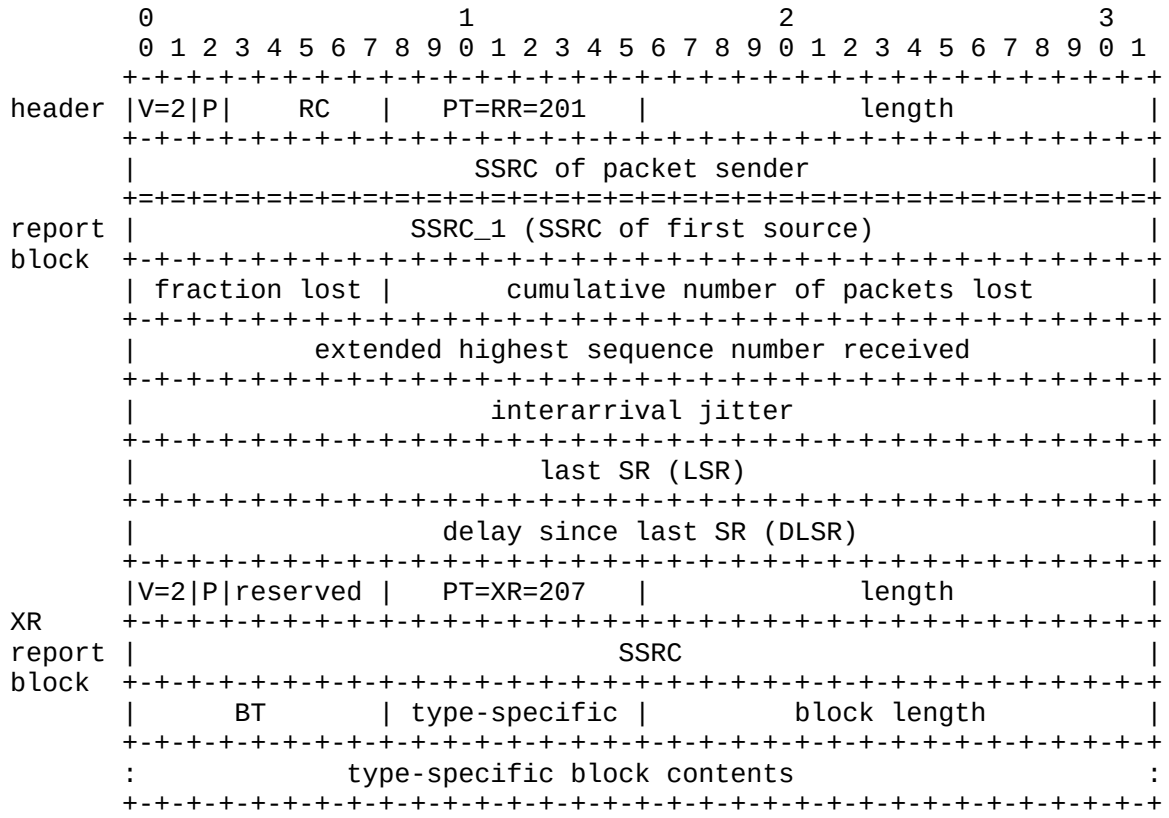
```
SETUP rtsp://example.com/foo/bar/baz.rm RTSP/1.0
    CSeq: 1
    OIF-QoS-Metrics: url:"rtsp://example.com/foo/bar/baz.rm";//
                        cumul-metrics=OIF-BasicPerfMonCumulSubset1;rate=2
```

The OITF would send the following message as a response to a GET_PARAMETER request:

OITF→CDF:

```
RTSP/1.0 200 OK
    CSeq: 15
    Session: 13320
    OIF-QoS-Feedback: url:"rtsp://example.com/foo/bar/baz.rm";//
                        PacketsDiscarded={125};PacketsOutOfSequence={20};//
                        PacketsReceived={2651};PacketsLost={7};DecodedFrames={1034}; //
                        LostFrames={2};DecodingErrors={25}
```

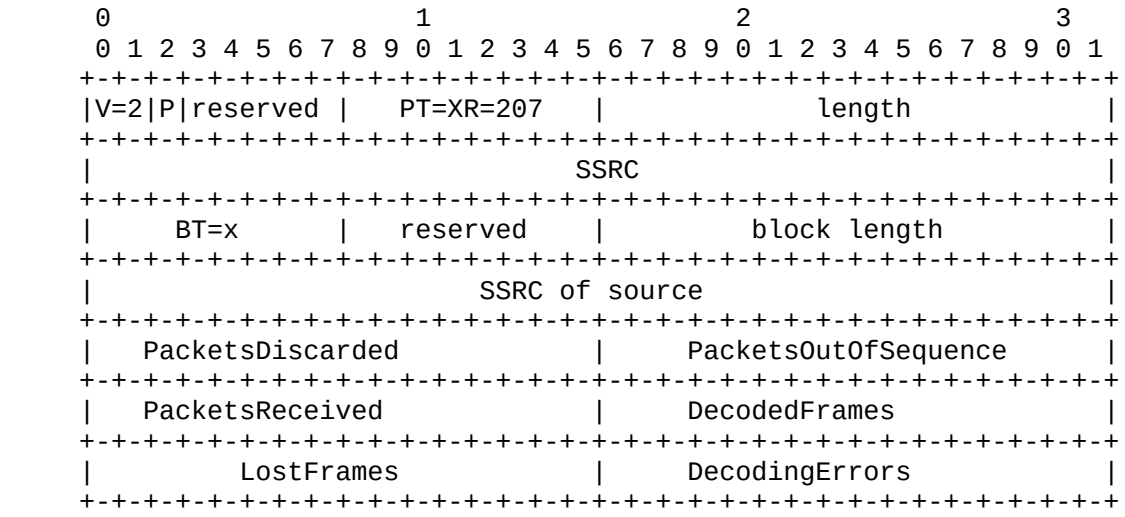
Open IPTV Forum specific sample metrics are reported using Extended Report blocks (XR) as per RFC 3611 **[RTCP-XR]**. These blocks are appended to the RTCP Receiver Reports, and may contain transport layer as well as application layer sample metrics. The RTCP Receiver Report including a XR extended report block would look as follows:



The following basic metrics are selected from TR-135 **[TR135]** as `OIPF-BasicPerfMonSampleSubset1` (metrics are same as for cumulative reporting but from different TR-135 objects):

- from the .STBService.{i}.ServiceMonitoring.MainStream.{i}.Sample.RTPStats object:
  - PacketsDiscarded, or late packets
  - PacketsOutOfSequence, or reordered packets
  - PacketsReceived and,
  - PacketsLost, which is equal to the value of “cumulative number of packets lost” in the RTCP Receiver Report.
- from the .STBService.{i}.ServiceMonitoring.MainStream.{i}.Sample.VideoDecoderStats object:
  - DecodedFrames and,
  - LostFrames
- from the .STBService.{i}.ServiceMonitoring.MainStream.{i}.Sample.AudioDecoderStats:
  - DecodingErrors

The RTCP XR packet will be:



where

- block length: 16 bits

  The length of this report block, including the header, in 32-bit words minus one. If the block type definition permits, zero is an acceptable value, signifying a block that consists of only the BT, type-specific, and block length fields, with a null type-specific block contents field.

- SSRC of the source:

  The SSRC of the RTP data packet source being reported upon by this report block.

Note: *The value of the Block Type (BT) is currently set to undefined, or "x". This value is to be set according to the value allocated by IANA for each set of metrics specified.*

B.2.5 Non-native HNI-IGI

The following figure illustrates the startup/initialization phase. The steps are outlined in detail below the figure.

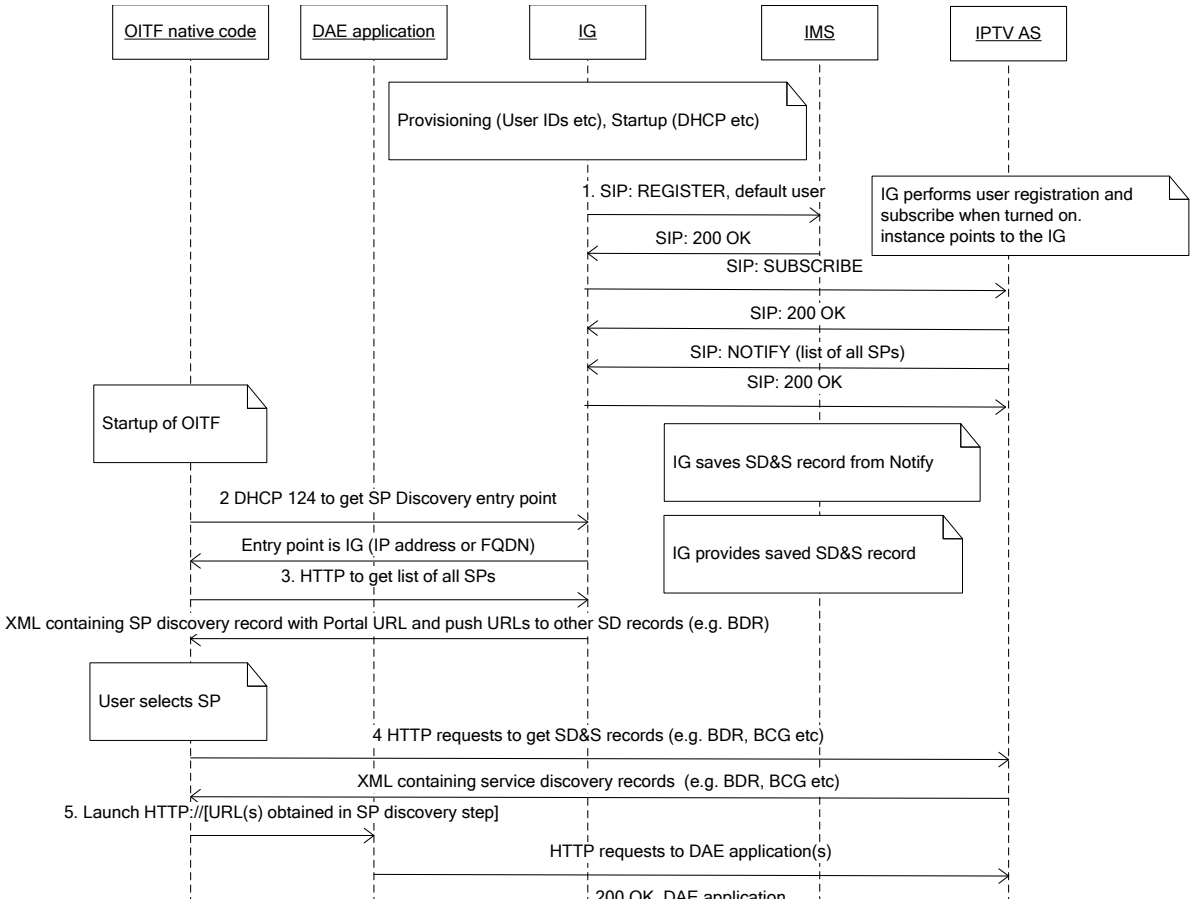


Figure 12: Registration for non-native HNI-IGI

As an initial condition of this example, it is assumed that the IG has been provisioned with appropriate information, e.g. User IDs, DHCP options have been carried out, etc.

Step 1: First, the IG performs SIP REGISTER for the default user. The instance ID used points to the IG, so that it is clear that this has no binding to OITFs. No applications are registered. After the registration, the IG performs SUBSCRIBE to get Service Provider Discovery information. These SD&S records are delivered in the SIP NOTIFY. The IG saves these records for later delivery to OITF.

Step 2: When the OITF is turned on it performs the normal startup procedure for an unmanaged device. Part of this startup is to perform DCHP option 124 in order to get Service Providers Discovery Entry points. The IG acts as a DHCP server and returns its own IP address in a DHCP option 125 response.

Step3: The OITF takes this IP address and makes an HTTP request to retrieve Service Providers Discovery information. The IG returns the XML structure it previously stored from the SIP NOTIFY.

Step 4: The XML structure obtained in the previous step contains information where to get Service Discovery information. This can be Web applications, HTTP servers, or multicast channels. In this flow http is assumed bu this is not a limitation. The OITF retrieves relevant discovery records, e.g. BDR (Broadcast Discovery Records). The unmanaged device support only OIP and hence BCG is optional. If the OITF supports BCG it SHALL fetch it (likely using multicast).

Step 5: When all the discovery records have been obtained, the OITF launches the discovered DAE applications.

Step 6: At least one of the DAE applications supports HNI-IGI in JavaScript. This DAE application sends a Pending IG request in order to receive unsolicited messages from IG/IMS. XMLHTTPRequest is used for HNI-IGI communication.

Step 7: The DAE application performs the normal HNI-IGI procedures as defined in the protocol specification. This includes registering users and applications, starting Scheduled content service, VoD etc. This is up to the DAE application.

Note that in the above call flow there are no changes required for the OITF, it acts just as an unmanaged OITF would normally do. There are some minor additions in order to support non-native HNI-IGI applications. Primarily these are methods in DAE to retrieve or set information in the OITF from DAE. In the protocol specification it is already defined that most applications can be DAE applications, and thus most of the methods required for non-native HNI-IGI is required independent of the non-native HNI-IGI.

B.3 Communication Services

B.3.1 Instant Messaging

B.3.1.1 Originating Instant Messages

Instant messaging uses paging mode, therefore it requires a session to be established between the 2 peers.

Figure 13 shows a call flow for an IPTV end-user invoking the Instant Messaging service. Below is a brief description for the call flow:

Step 1: The user invokes the instant messaging option on the OITF.

Step 2: The OITF issues a message request to the IG (See section 5.4.2.1.1, “Procedure for OITF Originating an Instant Messaging.”)

Step 3: The IG validates the request. (See section 6.4.2.1, “Procedure for Instant Messaging on UNIS-8.”)

Step 4: The IG issues a SIP MESSAGE to the messaging sever. (See section 6.4.2.1, “Procedure for Instant Messaging on UNIS-8.”)

Step 5: The server accepts the request with a SIP 200 OK response.

Step 6: The IG returns an HTTP 200 OK response to the HTTP POST that includes the SIP 200 OK response to the SIP MESSAGE

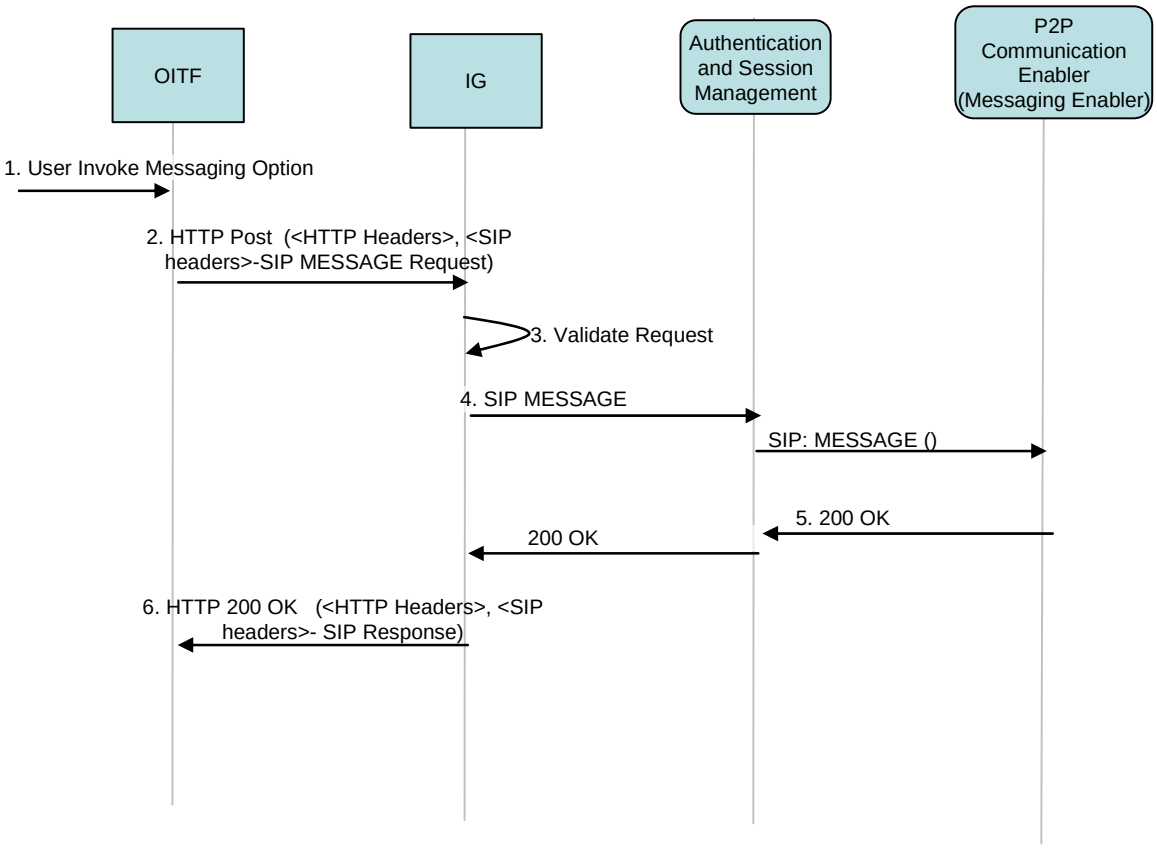


Figure 13: Instant Message Origination Call Flow

B.3.1.2 Incoming Instant Messages to IPTV end-users

Figure 14, shows a call flow for an incoming instant message to an IPTV end- user. Below is a brief description for the call flow:

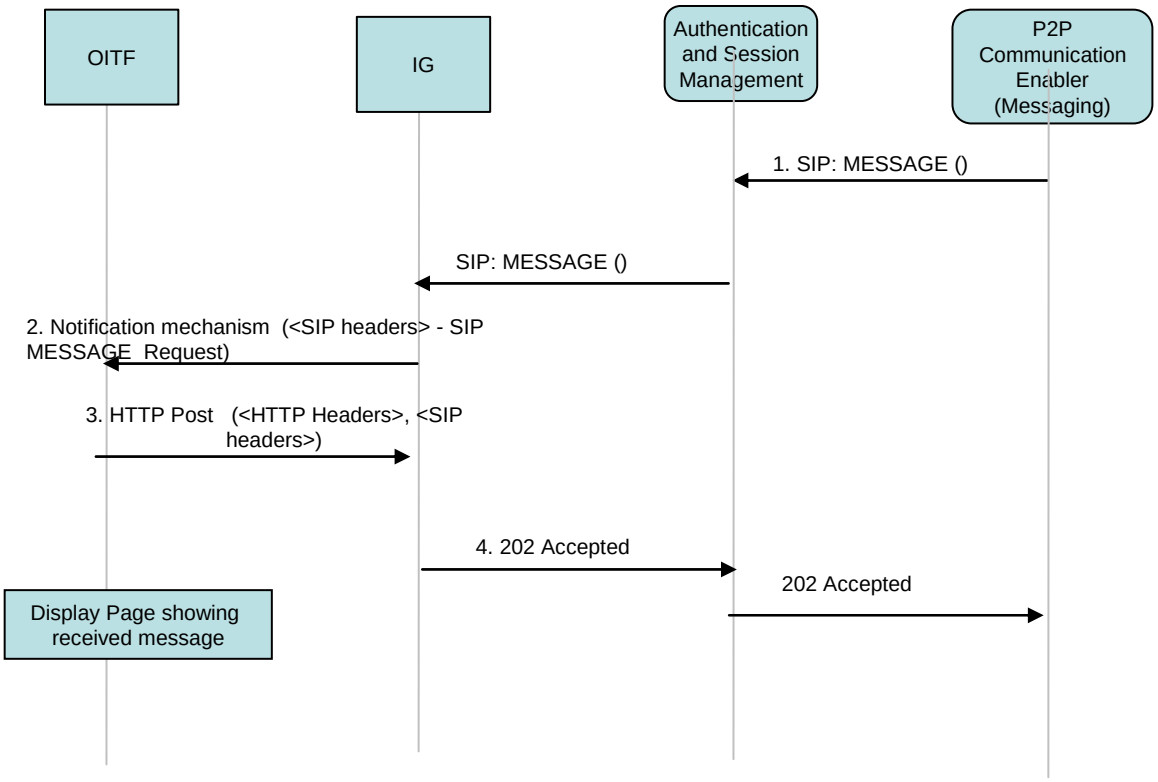


Figure 14: Incoming Message Call Flow

Step 1: The IG receives an incoming SIP MESSAGE

Steps 2-4: The IG forwards the SIP MESSAGE to the OITF (See section 5.4.2.1.2, “Incoming Instant Messaging Procedure.”)

B.3.2 Caller ID

B.3.2.1 Caller ID as a DAE or Embedded Application

Caller ID is identical to an incoming message to an IPTV end user. See section 5.4.1.1, “Procedure for Instant Message Based Caller ID” and section 6.4.2.1, “Procedure for Instant Messaging on UNIS-8” for details.

Figure 15 shows a call flow for Caller ID

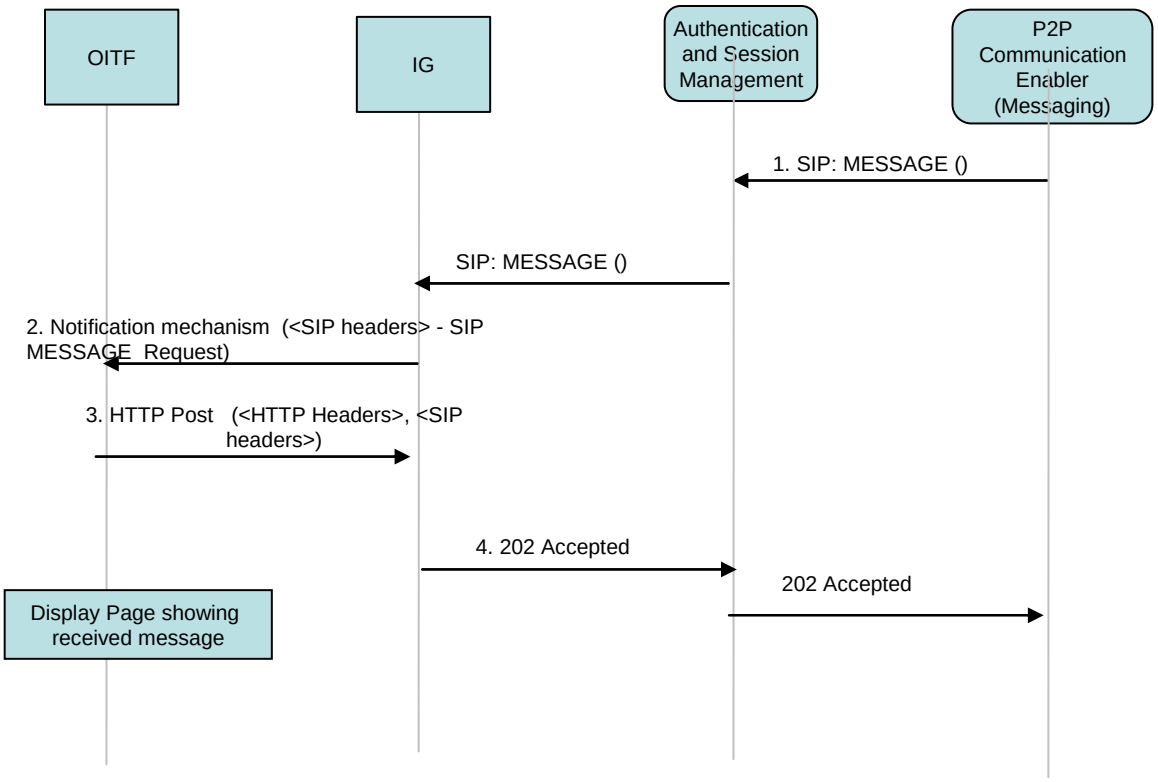


Figure 15: Caller identification Call Flow

B.3.2.2 Communication Services – Telephony service (Caller identification) for an incoming IMS voice call.

The IPTV solution provides a mechanism to enable the presentation of information on incoming IMS voice calls.

The managed networks, such as IMS, provide capability to connect multiple end-user terminals to communication services such as telephony service. The IMS Gateway, while registered to the IMS network, may also receive incoming SIP voice sessions and indicate the related information to the end-user.

The following figure shows a call flow with a caller identification based on the regular SIP INVITE request that is forwarded in parallel to the IMS Gateway and to a voice capable SIP/IMS UE. The IG gateway forwards the request towards the OITF, and also responds to the request with proper SIP response messages. The OITF gives a suitable indication to the end-user and the end-user can answer the incoming voice session using any of their voice capable clients connected to the IMS network.

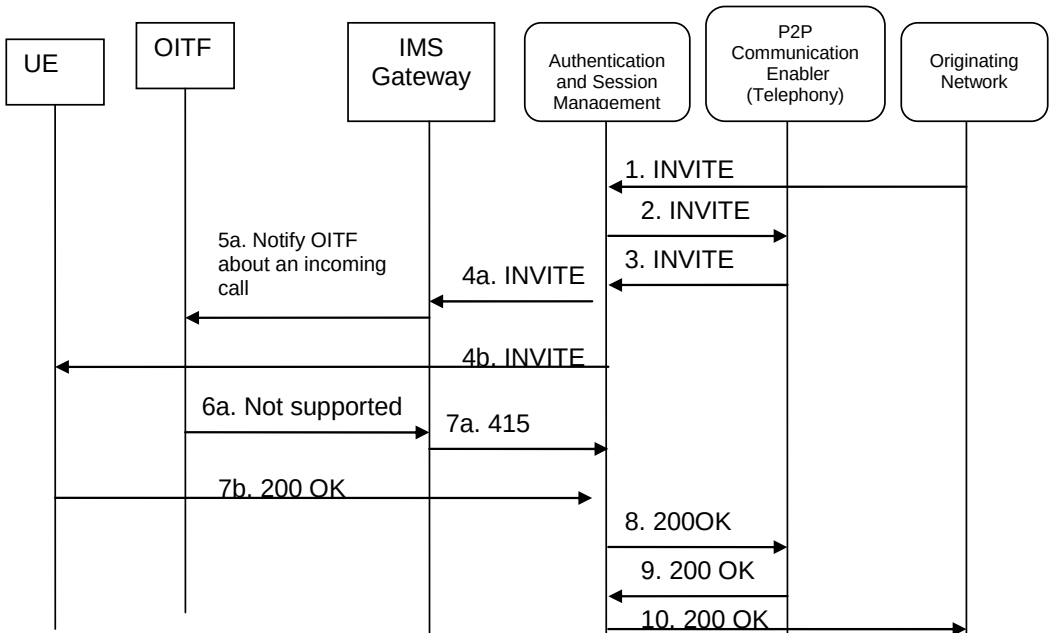


Figure 16: IMS telephony service based caller identification

The following procedure is supported in the OITF for caller identification

Step 1: The incoming voice session is forwarded to the end-user's IMS provider.

Step 2: Based on the initial filter criteria evaluation, the request is routed to the P2P communication enabler (Telephony service) in order to inform the P2P communication enabler of the incoming call.

Step 3: The request is routed back to the IMS network.

Step 4: Based on the user's terminal(s)´ registration information, configuration and terminal(s) capabilities, the session is routed to one or more user end devices. In this example, the end user has a voice capable SIP/IMS UE and the IMS Gateway registered with the same public user identity. A parallel forking is used and the INVITE is routed to:

4a. to the IMS gateway.

4b. and to the voice capable SIP UE.

Step 5a: The IMS Gateway sends a notification of the incoming call to the OITF. The following information can be presented to the OITF user:

- **Session Originator:** extracted from the P-Asserted-Identity header
- **Called party information:** indicates the called party, extracted from the P-Called-Party-ID header

The above parameters are based on SIP/SDP headers in the SIP INVITE request based RFC 3261 **[SIP]** and RFC 3455 **[RFC3455]**.

Step 6a: The OITF answers and replies with an indication that a voice call is not supported.

Step 7a: Response to the session setup is sent from the IMS GW to the IMS network (e.g. 415 Unsupported Media Type). The IMS network does not continue the dialog with the OITF but waits for the response from the SIP UE.

Step 7b: Parallel to the request 7a, the voice capable SIP UE answers the call and sends a 200 OK reply to the IMS network. Note that the normal session setup may include other SIP responses, e.g. 183 Session Progress; however, these are not shown here.

Step 8: The 200 OK is routed to the P2P communication enabler.

Step 9: The 200 OK is routed back to the IMS network.

Step 10: The 200 OK is routed back to the originating network.

B.3.3 Presence

B.3.3.1 End User Presence Services

The following is a list of Presence related services available to an IPTV end user:

- Subscription to Presence for one or multiple targets
- Cancellation of Presence subscription for one or multiple targets
- Publishing presence information related to an IPTV end user

B.3.3.2 Subscription to Presence

Figure 17 shows a call flow for an IPTV end- user subscription to Presence. Below is a brief description of the call flow:

Step 1: The procedure can be triggered by the user, through a menu selection, invoking the Presence option.

Step 2: The OITF issues a request to subscribe to Presence. See section 5.4.4.1, “Procedures for Subscription to Presence on the HNI-IGI.”).

Step 3: The IG validates that the request includes all the mandatory SIP headers for the subscription process. The IG rejects a request that does not include all mandatory SIP headers.

Step 4: The IG issues a SIP SUBSCRIBE to the Presence sever. See section 6.4.3.1, “Procedure for Presence on UNIS-8.”

Step 5: The server accepts the request with a SIP 200 OK response.

Step 6: The IG returns an HTTP 200 OK response to the HTTP POST (step 2) that includes the SIP 200 OK response to the SIP SUBSCRIBE.

Steps 7-10: These steps show the mechanism for OITF to receive the NOTIFY message. See section 5.4.4.1, “Procedures for Subscription to Presence on the HNI-IGI.”

Step 11: The IG forwards the SIP 200 OK to the network. Any subsequent notification messages incoming to the OITF shall be included in a HTTP 200 OK response to the HTTP POST in step 10.

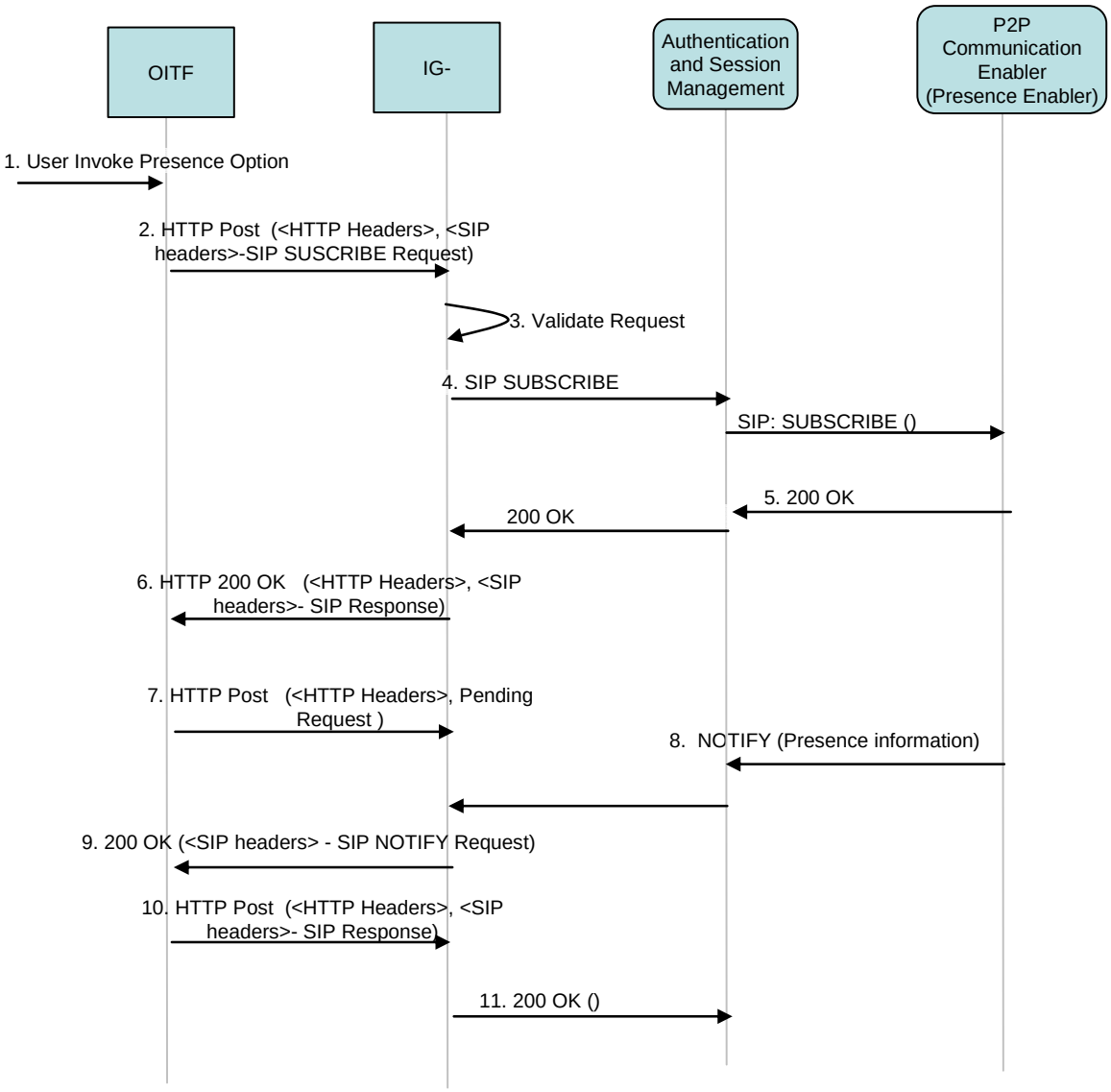


Figure 17: Subscription to Presence

B.3.3.3 Cancellation of Presence Subscription

Figure 18 shows a call flow for an IPTV end- user cancellation to an existing subscription to Presence. Below is a brief description of the call flow:

Step 1: The procedure can be triggered by the user, through a menu selection, invoking the Presence option.

- The OITF issues to cancel the presence subscription. (See section 5.4.4.2, “Procedure for Cancellation of a Subscription to Presence on the HNI-IGI.”)

Step 2: The IG validates that the request includes all the mandatory SIP headers for the subscription process. The IG rejects a request that does not include all mandatory SIP headers.

Step 3: The IG issues a SIP SUBSCRIBE with an Expiry time of 0 to the Presence sever.

Step 4: The server accepts the request with a SIP 200 OK response.

Step 5: The IG returns an HTTP 200 OK response to the HTTP POST that includes the SIP 200 OK response to the SIP SUBSCRIBE

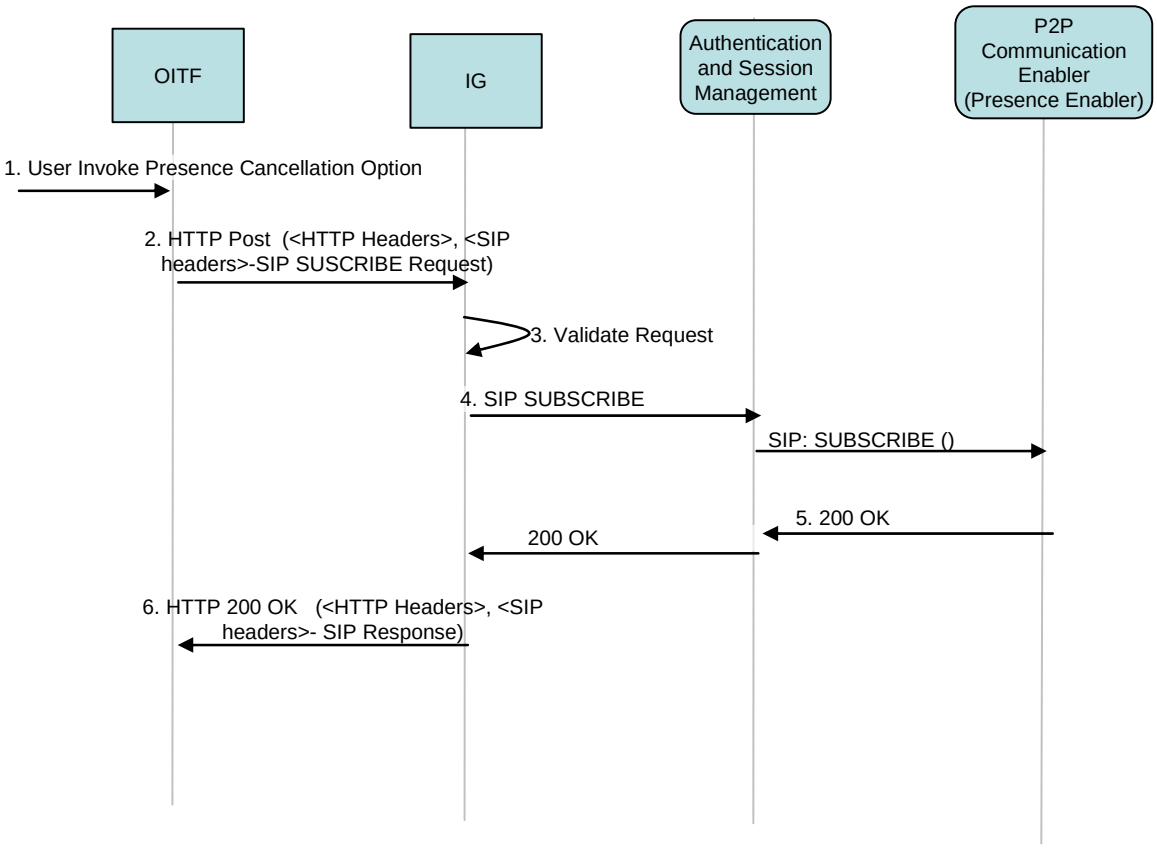


Figure 18: Cancellation of Presence Subscription

B.3.3.4 Publishing Presence Information

Figure 19 shows a call flow for an OITF publishing a Presence event to a server. Below is a brief description of the call flow:

Step 1: The OITF publishes presence information to the IG. See section 5.4.4.4, “Procedure for Publishing Presence information.”

Step 2: The IG validates that the request includes all the mandatory SIP headers for the publication process. The IG rejects a request that does not include all mandatory SIP headers.

Step 3: The IG issues a SIP PUBLISH to the Presence sever.

Step 4: The server accepts the request with a SIP 200 OK response.

Step 5: The IG returns an HTTP 200 OK response to the HTTP POST (from step 1) that includes the SIP 200 OK response to the SIP PUBLISH

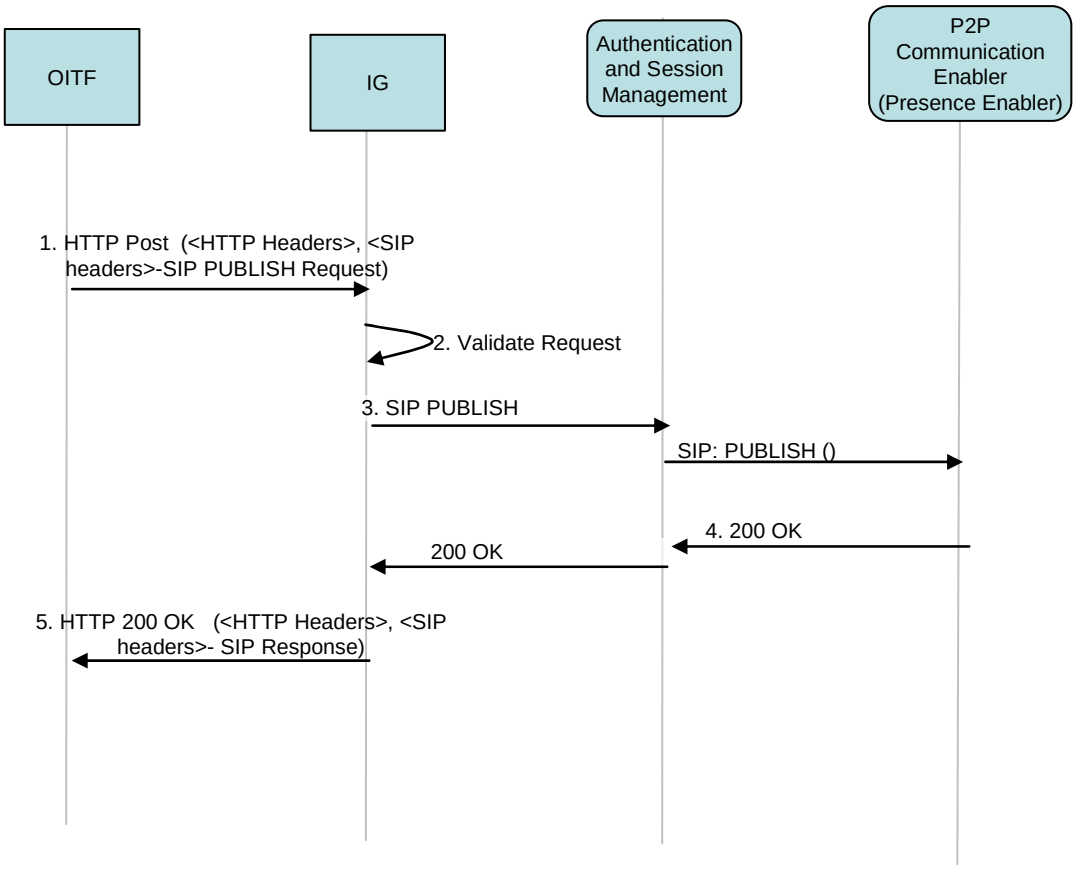


Figure 19: Publishing a Presence Event

Annex C Example Messages (informative)

C.1 IPTV Service Functions Message Examples

C.1.1 Example Messages for CoD session setup in a Managed Network

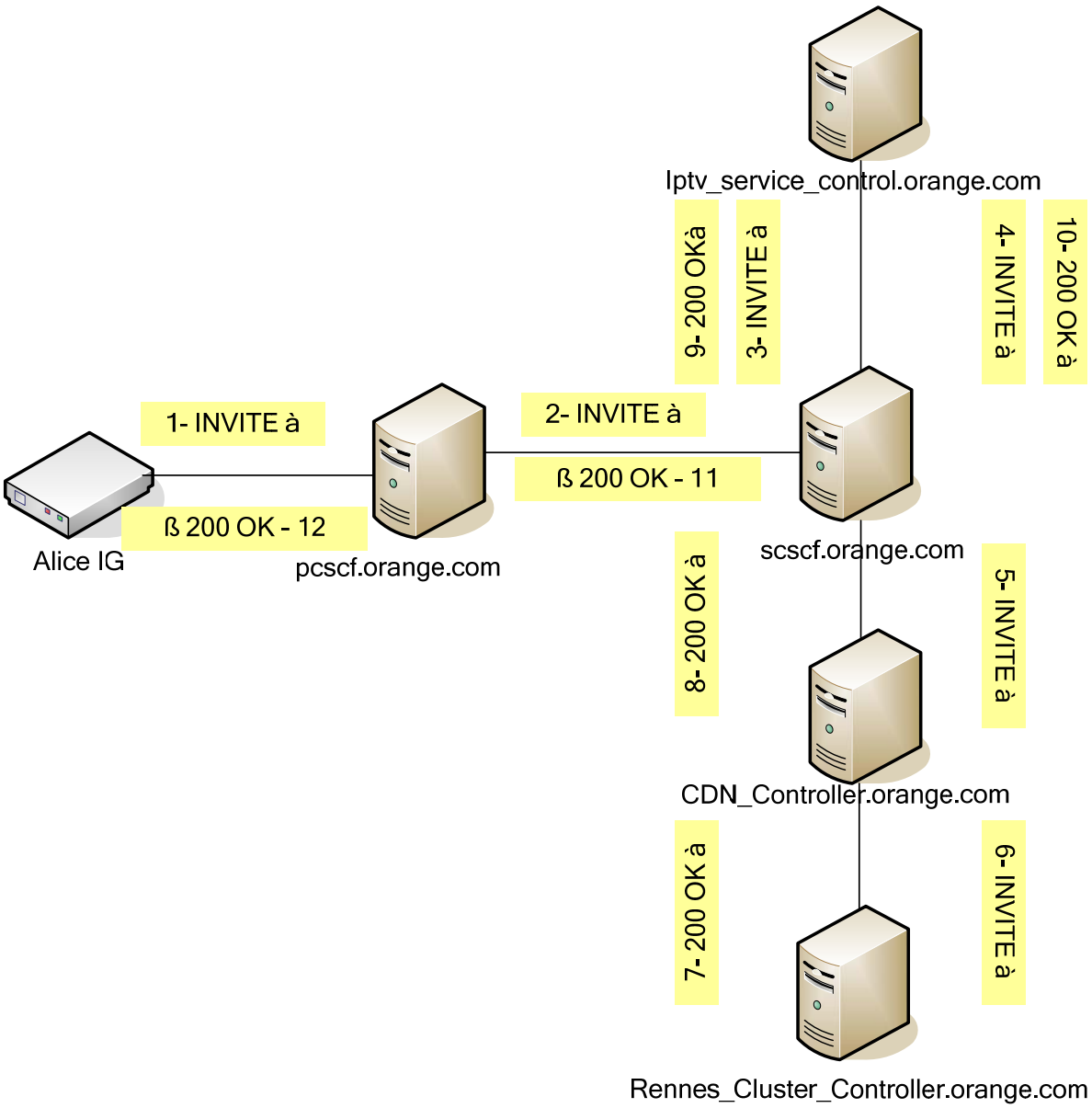


Figure 20: COD Session Set Up Sequence

Note: The IG had received in the response to the REGISTER the service route e.g. <sip:pcscf.orange.com:5060;lr;comp=sigcomp> <sip:scscf.orange.com:5060;lr; comp=sigcomp>

The following Request_URI example is for the HD version of the movie “Twister”. In the BCG, Twister is signalled with the CRID: “CRID://warnerbros.com/Twister” and the HD instance is signalled with the IMI: “imi:HD”.

Note: One or more of the characters ".", "/" and "#" in the example below may need to be escaped as %2E ("."), %2F ("/") and %23 ("#") depending on the restrictions imposed by the SIP Request URI.

In the call flows below, unchanged information (after the = sign) in any step are left blank.

Step 1: Alice IG to P-CSCF:

```
INVITE sip: IPTV_COD_SERVICE_warnerbros.com/Twister#HD@orange.com SIP/2.0 //CRID before @
Via: SIP/2.0/UDP  172.102.12.5:5061 //where to send the response
Max-Forwards: 70
Route: <sip:pcscf.orange.com:5060;lr;comp=sigcomp>,
       <sip:scscf.orange.com:5060;lr;comp=sigcomp>
From: AliceIG <sip:aliceIG@orange.com>;tag=1928301774 // tag for integrity verification
To: sip: IPTV_COD_SERVICE_warnerbros.com/Twister#HD@orange.com // same as request URI
CSeq: 314159 INVITE
Contact: <sip: 172.102.12.5:5061;transport=UDP>
Content-Type: application/sdp
Content-Length: (..)

v=0
o=AliceIG 2890844527 2890844527 IN IP4 172.102.12.5
s=Streaming Session
i=A Streaming session declared within the session description protocol
t=0
m=application 9 TCP iptv_rtsp // media line for RTSP control protocol
c=IN IP4 172.102.12.5
a=connection:new
a=setup:active

m=video 6666 RTP/AVP 33 // video
c=IN IP4 172.102.12.5
b= AS:4000
a=rcvonly
```

Step 2: P-CSCF to S-CSCF in ASM

```
INVITE
Via: SIP/2.0/UDP pcscf.orange.com:5060, SIP/2.0/UDP  172.102.12.5:5061
Max-Forwards: 69
Route: <sip:scscf.orange.com:5060;lr>
Record-Route: <sip:pcscf.orange.com:5060;lr;comp=sigcomp>
From:
To:
CSeq:
Contact:
Content-Type:
Content-Length: (..)

v=
o=
s=
i=
t=
m=
c=
a=
a=

m=
c=
b=
a=
```


Step 3: S-CSCF to IPTV Control

```
INVITE
Via: SIP/2.0/UDP scscf.orange.com:5060, SIP/2.0/UDP pcscf.orange.com:5060,
       SIP/2.0/UDP  172.102.12.5:5061,
Max-Forwards: 68
Record-Route: <sip:scscf.orange.com:5060;lr,comp=sigcomp>
       <sip:pcscf.orange.com:5060;lr;comp=sigcomp>
From:
To:
CSeq:
Contact:
Content-Type:
Content-Length: (..)
v=
o=
s=
i=
t=
m=
c=
a=
a=

m=
c=
b=
a=
```

Steps 4-5: IPTV Control to CDN Controller via S-CSCF

```
INVITE sip: CDN_Controller@orange.com SIP/2.0

Max-Forwards: 66
Route: <sip:CDN_Controller.orange.com:5060;lr>
Record-Route: <sip:scscf.orange.com:5060;lr;comp=sigcomp>,
       <sip:IPTV_SCSCF.orange.com:5060;lr;comp=sigcomp>,
       <sip:scscf.orange.com:5060;lr;comp=sigcomp>,
       <sip:pcscf.orange.com:5060;lr;comp=sigcomp>

From:
To:
CSeq:
Contact:
Content-Type:
Content-Length: (..)
v=
o=
s=
i=
t=
m=
c=
a=
a=

m=
c=
b=
a=
```

Step 6: CDN Controller to Rennes Cluster Controller

```
INVITE sip:Rennes_Cluster_Controller@orange.com SIP/2.0

Record-Route: <sip:CDN_Controller.orange.com:5060;lr,comp=sigcomp>,
       <sip:scscf.orange.com:5060;lr,comp=sigcomp>,
       <sip:IPTV_SCSCF.orange.com:5060;lr,comp=sigcomp>,
       <sip:scscf.orange.com:5060;lr, comp=sigcomp>,
       <sip:pcscf.orange.com:5060;lr;comp=sigcomp>
Max-Forwards: 65
Route: <sip:Rennes_Cluster_Controller.orange.com;lr>

From:
To:
CSeq:
Contact:
Content-Type:
Content-Length: (..)
v=
o=
s=
i=
t=
m=
c=
a=
a=
m=
c=
b=
a=
```

Step 7: Cluster Controller Reply to the CDN Controller

```
SIP/2.0 200 OK

Via: SIP/2.0/UDP CDN_Controller.orange.com
Record-Route:  <Rennes_Cluster_Controller.orange.com;lr,comp=sigcomp>

From:
To:
CSeq:
Contact:
Content-Type:
Content-Length: (..)

v=0
o= Rennes_Cluster_Controller IN IP4 Rennes_Cluster_Controller.orange.com
s=
i=
c=IN IP4 Rennes_Cluster_Controller.orange.com
t=0

m=application 999 TCP iptv_rtsp // media line for RTSP control protocol chosen by CC
a=connection:new
a=setup:passive
a=fmtp:iptv_rtsp h-uri=rtsp://Rennes_Cluster_Controller.orange.com/Twister ;
       h-session = 940211290776250

m=video 7777 RTP/AVP 33 // server video port
a=sendonly
```

Steps 8-12: 200 OK sent back to Alice IG using the same route

C.2 Communication Services Message Examples

C.2.1 Examples of HNI-IGI Message mapping to SIP

The sample mappings provided in this section are focused on Chat and Presence. They are not exhaustive.

The main use cases described in the Open IPTV Functional Architecture document are covered:

- Presence
  - Publication
  - Subscription
  - Notification
- Chat
  - Init
  - Outgoing message (standard)
  - Outgoing message (isComposing)
  - Incoming message
  - Teardown

As an illustration, a typical IMS presence document is also presented at the end of the section.

C.2.1.1 Presence

C.2.1.1.1 Initial publication

HNI-IGI Interface	**SIP equivalent**
POST *IG_URI/*SIP X-OITF-Request-Line: PUBLISH sip:david@oiptv.org SIP/2.0 Host : 192.168.1.1 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Expires: 3600** **X-OITF-Event: presence** **X-OITF-Call-Id:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x X-OITF-CSeq: 1 PUBLISH X-OITF-Content-Type: application/pidf+xml X-OITF-Content-length: (...) Content-Type: application/pidf+xml Content-Length: (...) <?xml version='1.0' encoding='UTF-8' ?> <presence xmlns='urn:ietf:params:xml:ns:pidf' entity='sip:david@oiptv.org'> ...	PUBLISH sip:david@oiptv.org SIP/2.0 Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=48240713 **To: sip:david@oiptv.org** CSeq: 1 PUBLISH Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Max-forwards: 10 **Expires: 3600** **Event: presence** Content-Type: application/pidf+xml Content-Length: (...) <?xml version='1.0' encoding='UTF-8' ?> <presence xmlns='urn:ietf:params:xml:ns:pidf' entity='sip:david@oiptv.org'> ... </presence>

</presence>	

HTTP/1.1 200 OK **X-OITF-Response-Line**: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Call-ID:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF- CSeq**: 1 PUBLISH **X-OITF- SIP-ETag**: 1514024804 **X-OITF- Expires**: 6002 X-OITF-Content-Length: 0 Content-Length: 0	SIP/2.0 200 OK Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=48240713 **To: sip:david@oiptv.org**;tag=12ba5d-287-55-1366522802 CSeq: 1 PUBLISH Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Expires: 6002 SIP-ETag: 1514024804 Content-Length: 0

Notes:

Specifying both a ‘From’ and a ‘To’ allows an identity to publish on behalf of another identity

Here the server has requested a shorter expiration time that will be handled internally by the IG

C.2.1.1.2 Updated publication

HNI-IGI Interface	**SIP equivalent**
POST *IG_URI/*SIP X-OITF-Request-Line: PUBLISH sip:david@oiptv.org SIP/2.0 Host : 192.168.1.1 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Expires: 3600** **X-OITF-Event: presence** **X-OITF-CSeq**: 2 PUBLISH **X-OITF-Call-ID**: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-SIP-if-match**: 15140248043 **X-OITF-Content-Type**: application/pidf+xml **X-OITF-Content-length**: (...) Content-Type: application/pidf+xml Content-Length: (...) <?xml version='1.0' encoding='UTF-8' ?> <presence xmlns='urn:ietf:params:xml:ns:pidf' entity='sip:david@oiptv.org'> ... </presence>	PUBLISH sip:david@oiptv.org SIP/2.0 Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=48240713 **To: sip:david@oiptv.org** CSeq: 2 PUBLISH Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Max-forwards: 10 SIP-if-match: 15140248043 **Expires: 3600** **Event: presence** Content-Type: application/pidf+xml Content-Length: (...) <?xml version='1.0' encoding='UTF-8' ?> <presence xmlns='urn:ietf:params:xml:ns:pidf' entity='sip:david@oiptv.org'> ... </presence>

HTTP/1.1 200 OK **X-OITF-Response-Line**: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Call-ID:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-CSeq**: 2 PUBLISH **X-OITF-SIP-ETag**: 7816034523 **X-OITF-Expires**: 600 **X-OITF-Content-Length:** 0 Content-Length: 0	SIP/2.0 200 OK Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=48240713 **To: sip:david@oiptv.org**;tag=12ba5d-287-55-1366522802 CSeq: 2 PUBLISH Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Expires: 600 SIP-ETag: 7816034523 Content-Length: 0

Notes:

SIP-IF-Match As retrieved from the previous publication acknowledgment

C.2.1.1.3 End of publication

HNI-IGI Interface	**SIP equivalent**
POST *IG_URI/*SIP Host : 192.168.1.1 X-IOTF-Request-Line: PUBLISH sip:david@oiptv.org SIP/2.0 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Expires: 0** **X-OITF-Event: presence** **X-OITF-CSeq**: 3 PUBLISH **X-OITF-Call-ID**: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-SIP-if-match**: 78160345234 **X-OITF-Content-Type:** application/pidf+xml **X-OITF-Content-length**: 0 Content-Type: application/pidf+xml Content-Length: 0	PUBLISH sip:david@oiptv.org SIP/2.0 Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=48240713 **To: sip:david@oiptv.org** CSeq: 3 PUBLISH Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Max-forwards: 10 SIP-if-match: 78160345234 **Expires: 0** **Event: presence** Content-Type: application/pidf+xml Content-Length: 0

HTTP/1.1 200 OK **X-OITF-Response-Line**: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Call-ID:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-CSeq**: 3 PUBLISH **X-OITF-Content-Length**: 0 Content-Length: 0	SIP/2.0 200 OK Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=48240713 **To: sip:david@oiptv.org**;tag=12ba5d-287-55-1366522802 CSeq: 3 PUBLISH Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Content-Length: 0

Notes:

Sip-if-match as retrieved from the previous publication acknowledgment

C.2.1.1.4 Initial subscription

HNI-IGI Interface	SIP equivalent
POST *IG_URI/*SIP **X-OITF-Request-Line**: SUBSCRIBE sip:fouz@oiptv.org SIP/2.0 Host : 192.168.1.1 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:fouz@oiptv.org** **X-OITF-Expires: 3600** **X-OITF-Event: presence** **X-OITF-Accept: application/pidf+xml** **X-OITF- CSeq**: 4 SUBSCRIBE **X-OITF- Call-ID**: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF- Content-length**: 0 Content-length: 0	SUBSCRIBE sip:fouz@oiptv.org SIP/2.0 Via: SIP/2.0/UDP 10.193.106.81:5060;branch=z9hG4bK1AD46E5D1E1 **From: sip:david@oiptv.org**;tag=2764425547 **To: sip:fouz@oiptv.org** CSeq: 4 SUBSCRIBE Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Contact: <sip:david@10.193.106.81:5060;transport=UDP> **Expires: 3600** **Event: presence** **Accept: application/pidf+xml** Content-length: 0

HTTP/1.1 200 OK **X-OITF-Response-Line**: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip: fouz@oiptv.org** **X-OITF-Call-ID:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-CSeq**: 4 SUBSCRIBE **X-OITF-Expires**: 6005 **X-OITF-Content-Length:** 0 Content-Length: 0	SIP/2.0 200 OK Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=2764425547 **To: sip:fouz@oiptv.org**;tag=12ba5d-287-55-1366522802 CSeq: 4 SUBSCRIBE Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Expires: 6005 Content-Length: 0

Notes:

Here the server has requested a shorter expiration time that will be handled internally by the IG

C.2.1.1.5 Notification (individual)

HNI-IGI Interface	SIP equivalent
HTTP 200 OK **X-OITF-Response-Line**: NOTIFY sip:david@10.193.106.72:5060;transport=UDP SIP/2.0 **X-OITF-From: sip:fouz@oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Event: presence** **X-OITF-CSeq**: 1001 NOTIFY **X-OITF-Call-ID**: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-Subscription-State:** active; expires=600 **X-OITF-Content-Type**: application/pidf+xml **X-OITF-Content-Length**: (...) Content-Type: application/pidf+xml Content-Length: (...) <?xml version="1.0" encoding="UTF-8"?> <presence xmlns="urn:ietf:params:xml:ns:pidf"	NOTIFY sip:david@10.193.106.72:5060;transport=UDP SIP/2.0 Via: SIP/2.0/UDP 10.194.117.18:5060;branch=z9hG4bK0014c262ba5d-2 **From: sip:fouz@oiptv.org**;tag=10014c262ba5d **To: sip:david@oiptv.org**;tag=2764425547 CSeq: 1001 NOTIFY Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Subscription-State: active; expires=600 Contact: <sip:10.194.117.18:5060;transport=UDP> **Event: presence** Content-Type: application/pidf+xml Content-Length: (...) <?xml version="1.0" encoding="UTF-8"?> <presence xmlns="urn:ietf:params:xml:ns:pidf"

entity="sip:fouz@oiptv.org"> ... </presence>	entity="sip:fouz@oiptv.org"> ... </presence>

Note that an acknowledgment from the IG to the network is required but not described here.

C.2.1.1.6 Subscription Refresh

HNI-IGI Interface	SIP equivalent
POST *IG_URI*/SIP X-OITF-Request-Line: SUBSCRIBE sip:fouz@oiptv.org SIP/2.0 Host: 192.168.1.1 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:fouz@oiptv.org** **X-OITF-Expires: 3600** **X-OITF-Event: presence** **X-OITF-Accept: application/pidf+xml** **X-OITF-CSeq**: 5 SUBSCRIBE **X-OITF-Call-ID**: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-Contact:** <sip:david@10.193.106.81:5060;transport=UDP> **X-OITF-Content-length**: 0 Content-length: 0	SUBSCRIBE sip:fouz@oiptv.org SIP/2.0 Via: SIP/2.0/UDP 10.193.106.81:5060;branch=z9hG4bK1AD46E5D1E1 **From: sip:david@oiptv.org**;tag=2764425547 **To: sip:fouz@oiptv.org** CSeq: 5 SUBSCRIBE Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Contact: <sip:david@10.193.106.81:5060;transport=UDP> **Expires: 3600** **Event: presence** **Accept: application/pidf+xml** Content-length: 0

HTTP/1.1 200 OK **X-OITF-Response-Line**: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip: fouz@oiptv.org** **X-OITF-Call-ID:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-CSeq**: 5 SUBSCRIBE **X-OITF-Expires**: 6005 **X-OITF-Content-Length:** 0 Content-Length: 0	SIP/2.0 200 OK Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=2764425547 **To: sip:fouz@oiptv.org**;tag=12ba5d-287-55-1366522802 CSeq: 5 SUBSCRIBE Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Expires: 6006 Content-Length: 0

Note that from the OITF's point-of-view, both initial and refresh subscription requests are identical.

Here the server has requested a shorter expiration time, which will be handled internally by the IG

C.2.1.1.7 End of subscription

HNI-IGI Interface	SIP equivalent
POST *IG_URI*/SIP **X-OITF-Request-Line**: SUBSCRIBE sip:fouz@oiptv.org SIP/2.0 Host: 192.168.1.1 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:fouz@oiptv.org** **X-OITF-Expires: 0**	SUBSCRIBE sip:fouz@oiptv.org SIP/2.0 Via: SIP/2.0/UDP 10.193.106.81:5060;branch=z9hG4bK1AD46E5D1E1 **From: sip:david@oiptv.org**;tag=2764425547 **To: sip:fouz@oiptv.org** CSeq: 6 SUBSCRIBE

HNI-IGI Interface	SIP equivalent
X-OITF-Event: presence **X-OITF-Accept: application/pidf+xml** **X-OITF**-CSeq: 6 SUBSCRIBE **X-OITF-Call-ID:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-Contact**: <sip:david@10.193.106.81:5060;transport=UDP> **X-OITF-Content-length**: 0 Content-length: 0	Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Contact: <sip:david@10.193.106.81:5060;transport=UDP> **Expires: 0** **Event: presence** **Accept: application/pidf+xml** Content-length: 0

HTTP/1.1 200 OK **X-OITF-Response-Line**: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip: fouz@oiptv.org** **X-OITF-Call-ID:** 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x **X-OITF-CSeq**: 5 SUBSCRIBE **X-OITF-Expires**: 6005 X-OITF-Content-Length: 0 Content-Length: 0	SIP/2.0 200 OK Via: SIP/2.0/UDP 10.194.56.134:5060;branch=z9h4bK61C529E16989 **From: sip:david@oiptv.org**;tag=2764425547 **To: sip:fouz@oiptv.org**;tag=12ba5d-287-55-1366522802 CSeq: 6 SUBSCRIBE Call-ID: 78A0g080Ca3502i6414m360Bt38A6b2E4Fx61C8x Content-Length: 0

C.2.1.2 Chat

C.2.1.2.1 Chat session setup

HNI-IGI Interface	SIP equivalent
POST *IG_URI*/SIP Host: 192.168.1.1 **X-OITF-Request-Line: INVITE** sip:sports.room@chat.oiptv.org SIP/2.0 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:sports.room@chat.oiptv.org** **X-OITF-Accept: message/cpim** **X-OITF-Call-ID: 3413an89KU** **X-OITF-**Content-Type: application/sdp **X-OITF-**Content-length: (...) Content-length: 0	INVITE sip:sports.room@chat.oiptv.org SIP/2.0 **To: sip:sports.room@chat.oiptv.org** **From: sip:david@oiptv.org**;tag=786 Call-ID: 3413an89KU Content-Type: application/sdp Content-length: (...) c=IN IP4 10.194.52.13 m=message 7654 TCP/MSRP * a=**accept-types:message/cpim** a=path:msrp://10.194.52.13:7654/jshA7weztas;tcp

HTTP/1.1 200 OK X-OITF-Response-Line: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:sports.room@chat.oiptv.org**	SIP/2.0 200 OK **To: sip:sports.room@chat.oiptv.org**;tag=087js **From: sip:david@oiptv.org**;tag=786 Call-ID: 3413an89KU

X-OITF-Call-ID: 3413an89KU **X-OITF-**Content-Type: application/sdp **X-OITF-Accept: message/cpim** **X-OITF-**Content-length: (...) Content-Length: 0	Content-Type: application/sdp Content-length: (...) c=IN IP4 chat.oiptv.org m=message 12763 TCP/MSRP * a=accept-types:message/cpim a=path:msrp://chat.oiptv.org:12763/kjhd37s2s20w2a;tcp

Note that a final acknowledgment from the IG to the network is required, but not described here.

C.2.1.2.2 Chat outgoing message (standard)

HNI-IGI Interface	MSRP equivalent
POST *IG_URI*/AUX **X-HNI-IGI-Request: MSRP SEND MESSAGE** **X-HNI-IGI-Message-ID:** **X-HNI-IGI-From: sip:david@oiptv.org** **X-HNI-IGI-To: sip:sports.room@chat.oiptv.org** Who else thinks this late penalty was a disgrace ?	MSRP a786hjs2 SEND **To-Path: msrp://chat.oiptv.org:12763/kjhd37s2s20w2a;tcp8** **From-Path: msrp://10.194.52.13:7654/jshA7weztas;tcp8** **Message-ID: 87652491** Byte-Range: (...) Content-Type: message/cpim To: sip:sports.room@chat.oiptv.org From: David <sip:david@oiptv.org> DateTime: 2008-06-15T15:02:31-03:00 Content-Type: text/plain Who else thinks this late penalty was a disgrace ? -------a786hjs2$
HTTP/1.1 200 OK **X-HNI-IGI-Response-Line: MSRP 200 OK** **X-HNI-IGI-Message-ID: 87652491** **X-HNI-IGI-From: sip:david@oiptv.org** **X-HNI-IGI-To: sip:sports.room@chat.oiptv.org** Content-Length: 0	MSRP a786hjs2 200 OK To-Path: msrp://chat.oiptv.org:12763/kjhd37s2s20w2a;tcp From-Path: msrp://10.194.52.13:7654/jshA7weztas;tcp -------a786hjs2$

The IG is responsible for mapping the caller and callee URIs to the actual MSRP paths exchanged during the chat setup

C.2.1.2.3 Chat outgoing message (isComposing)

HNI-IGI Interface	MSRP equivalent
POST *IG_URI*/AUX X-HNI-IGI-Request: MSRP SEND ACTIVITY **X-OITF-Message-ID: 87653492** **X-HNI-IGI-From: sip:david@oiptv.org** **X-HNI-IGI-To: sip:sports.room@chat.oiptv.org** <?xml version="1.0" encoding="UTF-8"?>	MSRP a786hjs2 SEND **To-Path: msrp://chat.oiptv.org:12763/kjhd37s2s20w2a;tcp9** **From-Path: msrp://10.194.52.13:7654/jshA7weztas;tcp9** **Message-ID: 87652492**

<isComposing xmlns="urn:ietf:params:xml:ns:im-iscomposing" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:ietf:params:xml:ns:im-composing iscomposing.xsd"> <state>active</state> <contenttype>text/plain</contenttype> <refresh>90</refresh> </isComposing>	Byte-Range: (...) Content-Type: message/cpim Content-Length: (..) To: sip:sports.room@chat.oiptv.org From: David <sip:david@oiptv.org> DateTime: 2008-06-15T15:02:31-03:00 Content-Type: application/im-iscomposing+xml <?xml version="1.0" encoding="UTF-8"?> <isComposing xmlns="urn:ietf:params:xml:ns:im-iscomposing" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:ietf:params:xml:ns:im-composing iscomposing.xsd"> <state>active</state> <contenttype>text/plain</contenttype> <refresh>90</refresh> </isComposing> -------a786hjs2$

HTTP/1.1 200 OK **X-HNI-IGI-Response-Line: MSRP 200 OK** **X-HNI-IGI-Message-ID: 87652491** **X-HNI-IGI-From: sip:david@oiptv.org** **X-HNI-IGI-To: sip:sports.room@chat.oiptv.org** Content-Length: 0	MSRP a786hjs2 200 OK To-Path: msrp://chat.oiptv.org:12763/kjhd37s2s20w2a;tcp From-Path: msrp://10.194.52.13:7654/jshA7weztas;tcp -------a786hjs2$

The IG is responsible for mapping the caller and callee URIs to the actual MSRP paths exchanged during the chat setup

C.2.1.2.4 Chat incoming message

HNI-IGI Interface	**MSRP equivalent**
HTTP/1.1 200 OK X-HNI-IGI-Request: MSRP RECEIVED MESSAGE **X-OITF-From: sip:sports.room@chat.oiptv.org** **X-OITF-To: sip:david@oiptv.org** **X-OITF-Message-ID: 56712483** I don't care: we won anyway !	MSRP a786hjs2 SEND **To-Path: msrp://10.194.52.13:7654/jshA7weztas;tcp10** **From-Path: msrp://chat.oiptv.org:12763/kjhd37s2s20w2a;tcp10** **Message-ID: 56712483** Byte-Range: (...) Content-Type: message/cpim Content-Length: (…) To: sip:sports.room@chat.oiptv.org

HNI-IGI Interface	MSRP equivalent
	From: Fouz <sip:fouz@oiptv.org> DateTime: 2008-06-15T15:02:31-03:00 Content-Type: text/plain I don't care: we won anyway ! -------a786hjs2$

Note that an acknowledgment from the IG to the network is required, but not described here.

The IG SHALL be responsible for mapping the MSRP paths exchanged during the chat setup to the actual caller and callee URIs

C.2.1.2.5 Chat session teardown

HNI-IGI Interface	SIP equivalent
POST *IG_URI*/SIP **X-OITF-Request-Line:** BYE sip:sports.room@chat.oiptv.org SIP/2.0 Host: 192.168.1.1 **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:sports.room@chat.oiptv.org** **X-OITF-Call-ID: 3413an89KU11** **X-OITF-CSeq: 231 BYE** **X-OITF-Content-Length: 0** Content-length: 0	BYE sip:sports.room@chat.oiptv.org SIP/2.0 Via: SIP/2.0/UDP 192.0.2.4;branch=z9hG4bKnashds10 Max-Forwards: 70 **To: sip:sports.room@chat.oiptv.org**;tag=087js **From: sip:david@oiptv.org**;tag=786 Call-ID: 3413an89KU11 CSeq: 231 BYE Content-Length: 0

HTTP/1.1 200 OK X-OITF-Response-Line: SIP/2.0 200 OK **X-OITF-From: sip:david@oiptv.org** **X-OITF-To: sip:sports.room@chat.oiptv.org** **X-OITF-Call-ID: 3413an89KU** **X-OITF-CSeq: 231 BYE** **X-OITF-Content-Length: 0** Content-Length: 0	SIP/2.0 200 OK **To: sip:sports.room@chat.oiptv.org**;tag=087js **From: sip:david@oiptv.org**;tag=786 Call-ID: 3413an89KU CSeq: 231 BYE Content-length: 0

C.2.1.3 Presence Document

C.2.1.3.1 Presence Schema

See Annex I for the Presence XML Schema.

C.2.1.3.2 Presence schema examples

Examples of how the Presence information semantics are described in a typical Presence Information XML schema are shown below.

C.2.1.3.2.1 Example of Open IPTV Presence for Scheduled Content service

```
<?xml version="1.0" encoding="UTF-8"?>

<presence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:iptv="urn:oipf:service:oitfpresence:2008"
  xmlns:oma="urn:oma:xml:prs:pidf:oma-pres"
  xmlns:pdm="urn:ietf:params:xml:ns:pidf:data-model"
  xmlns="urn:ietf:params:xml:ns:pidf" entity="sip:someone@example.com">
  <import schemaLocation="service-presence.xsd" />
  <tuple id="abcde">
    <status>
      <basic>open</basic>
    </status>

    <oma:service-description>
      <oma:service-id>IPTV-BC</oma:service-id>
      <oma:version>1.0</oma:version>
      <oma:description>IPTV Scheduled Content</oma:description>
    </oma:service-description>

    <BC>
      <currentBCServiceID>BC-service-id</currentBCServiceID>
      <currentBCProgramID>BC-program-id</currentBCProgramID>
    </BC>

    <timestamp>2008-07-08T12:34:21Z</timestamp>
  </tuple>

  <pdm:device id="aa111">
    <pdm:deviceID>
      urn:uuid:11162e19-5fbf-43fc-a2fd-d23002787599
    </pdm:deviceID>
    <pdm:timestamp>2008-07-08T12:34:21Z</pdm:timestamp>
  </pdm:device>

</presence>
```

C.2.1.3.2.2 Example of Open IPTV Presence for Hybrid service

```
<?xml version="1.0" encoding="UTF-8"?>
<presence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:iptv="urn:oipf:service:oitfpresence:2008"
  xmlns:oma="urn:oma:xml:prs:pidf:oma-pres"
  xmlns:pdm="urn:ietf:params:xml:ns:pidf:data-model"
  xmlns="urn:ietf:params:xml:ns:pidf" entity="sip:someone@example.com">
  <import schemaLocation="service-resence.xsd" />
  <tuple id="abcde">
    <status>
      <basic>open</basic>
    </status>

    <oma:service-description>
      <oma:service-id>IPTV-Hybrid</oma:service-id>
      <oma:version>1.0</oma:version>
      <oma:description>IPTV Hybrid service</oma:description>
    </oma:service-description>
    <iptv:IPTVHybridService>
      <iptv:IPTVHybrid Tecnology="DVB-T">
        <iptv:watchedBroadcast>
          <iptv:currentChannel>BCC</iptv:currentChannel>
          <iptv:currentProgram>News</iptv:currentProgram>
          <iptv:serviceID>BBC_ID</iptv:serviceID>
          any
        </iptv:watchedBroadcast>
        any
      </iptv:IPTVHybrid>
    </iptv:IPTVHybridService>
```



```
    <timestamp>2008-07-08T12:34:21Z</timestamp>
  </tuple>

  <pdm:device id="aa111">
    <pdm:deviceID>
      urn:uuid:11162e19-5fbf-43fc-a2fd-d23002787599
    </pdm:deviceID>
    <pdm:timestamp>2008-07-08T12:34:21Z</pdm:timestamp>
  </pdm:device>

</presence>
```

Annex D User Profile Description (informative)

D.1 IPTV Subscription Profile

IPTV Subscription Profile encompasses relevant information required to operate an IPTV service. This includes user settings regarding:

- Global settings (Language preference, user action recordable).
- Broadcast settings, with List of subscribed BC service packages.

Broadcast service refers to Scheduled Content services. Accordingly, Broadcast settings refer to the Scheduled Content settings.

A BC service package is a set of elementary BC TV services, along with a description. These BC services have the same authorization and charging policy.

A BC IPTV service is for instance a multicast channel, interactive channel, mosaic that a user may subscribe to.

NOTE: The Broadcast settings only provide a reference to service package and/or associated services that a given IPTV user has subscribed to, and is not meant to be a complete description of the service package and/or service. The complete service package and/or service description SHOULD be available in an associated IPTV Service Profile definition. If the list of elementary IPTV services associated with a given service package are not explicitly listed in the IPTV subscription profile, then it implies that the IPTV user has implicitly subscribed to all of the IPTV services within that service package.

- Content on demand settings (Parental control level).
- PVR settings (PVR preferences network/local, PVR user restrictions, PVR storage limit).
- User Equipment information (OITF) which uniquely identifies the user's OITF, classifies it as a device type (OITF-STV, OITF-TV) and provides relevant device capabilities. An IPTV user may be associated with one or more OITF(s) and every OITF is uniquely identified with a Unique Identifier (tUEID). The OITF capabilities associated with an IPTV user profile may be used for customization of IPTV service selection data that is provided by the IPTV Service Discovery to the IPTV user (based on capabilities of the OITF with which the IPTV user is currently associated). For instance, an IPTV user on a SD-only device would not be provided with information related to HD services. The OITF settings is not intended to cover all information related to the OITF and currently holds only the OITF capabilities attribute since this information may be used by the IPTV service discovery for personalized service selection.

Note that detailed information about the OITF may be located elsewhere and can be referenced by the IPTV Subscription Profile using the tUEID element.

D.1.1 OITF XML Schema for the IPTV Subscription Profile

OITF XML Schema for the IPTV profile, based on **[TS183063]** Annex C:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="org:oipf:iptv:IPTVProfile:2008
  xmlns:tns="org:oipf:iptv:IPTVProfile:2008"
  xmlns:ueprofile="org:oipf:iptv:UEProfile:2008-1"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <import namespace="org:oipf:iptv:UEProfile:2008-1"
    schemalocation="iptv-UEProfile.xsd" />
  <element name="IPTVProfile">
    <annotation>
      <documentation>
        XML Schema for representing the IPTV Profile object
      </documentation>
    </annotation>
```



```
  <complexType>
   <sequence>
    <element name="UEProfile" type="ueprofile:tUEProfile" minOccurs="0" />
    <element name="GlobalSettings" type="tns:tGlobalSettings" />
    <element name="BCProfile" type="tns:tBCProfile" minOccurs="0" />
    <element name="CoDProfile" type="tns:tCoDProfile" minOccurs="0" />
    <element name="PVRProfile" type="tns:tPVRProfile" minOccurs="0" />
    <element name="Extension" type="tns:tExtension" minOccurs="0" />
    <any namespace="##other" processContents="lax" minOccurs="0"
     maxOccurs="unbounded" />
   </sequence>
   <attribute name="ProfileId" type="ID" />
   <anyAttribute/>
  </complexType>
 </element>

 <complexType name="tBCProfile">
  <sequence>
   <element name="BCServicePackage" type="tns:tBCServicePackage"
    maxOccurs="unbounded" />
   <element name="Extension" type="tns:tExtension" minOccurs="0" />
   <any namespace="##other" processContents="lax" minOccurs="0"
    maxOccurs="unbounded" />
  </sequence>
 </complexType>

 <complexType name="tBCServicePackage">
  <sequence>
   <element name="BCPackageId" type="tns:tBCServicePackageID"/>
   <element name="Description" type="tns:tBCServicePackageDescription"
    minOccurs="0" />
   <element name="BCService" type="tns:tBCService" minOccurs="0"
    maxOccurs="unbounded" />
   <element name="Extension" type="tns:tExtension" minOccurs="0" />
   <any namespace="##other" processContents="lax" minOccurs="0"
    maxOccurs="unbounded" />
  </sequence>
 </complexType>

 <simpleType name="tBCServicePackageID" final="list restriction">
  <restriction base="string">
   <minLength value="0" />
   <maxLength value="16" />
  </restriction>
 </simpleType>

 <simpleType name="tBCServicePackageDescription" final="list restriction">
  <restriction base="string">
   <minLength value="0" />
   <maxLength value="64" />
  </restriction>
 </simpleType>

 <complexType name="tBCService">
  <sequence>
   <element name="ParentalControl" type="tns:tParentalControlLevel"
    minOccurs="0" />
   <element name="BCServiceId" type="tns:tBCServiceID" minOccurs="1" />
   <element name="QualityDefinition" type="tns:tQualityDefinition"
    minOccurs="0" />
   <element name="Extension" type="tns:tExtension" minOccurs="0" />
   <any namespace="##other" processContents="lax" minOccurs="0"
```



```
      maxOccurs="unbounded"/>
  </sequence>
</complexType>

<simpleType name="tBCServiceID" final="list restriction">
  <restriction base="string">
    <minLength value="0" />
    <maxLength value="16" />
  </restriction>
</simpleType>

<simpleType name="tQualityDefinition" final="list restriction">
  <restriction base="unsignedByte">
    <minInclusive value="0" />
    <maxInclusive value="1" />
    <enumeration value="0">
      <annotation>
        <documentation>
          <label xml:lang="en">SD</label>
          <definition xml:lang="en">Standard Definition</definition>
        </documentation>
      </annotation>
    </enumeration>
    <enumeration value="1">
      <annotation>
        <documentation>
          <label xml:lang="en">HD</label>
          <definition xml:lang="en">High Definition</definition>
        </documentation>
      </annotation>
    </enumeration>
  </restriction>
</simpleType>

<complexType name="tCoDProfile">
  <sequence>
    <element name="ParentalControl" type="tns:tParentalControlLevel"
      minOccurs="0"/>
    <element name="Extension" type="tns:tExtension" minOccurs="0"/>
    <any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </sequence>
</complexType>

<simpleType name="tParentalControlLevel" final="list restriction">
  <restriction base="unsignedByte">
    <minInclusive value="0"/>
    <maxInclusive value="5"/>
    <enumeration value="0">
      <annotation>
        <documentation>
          <label xml:lang="en">ALL</label>
          <definition xml:lang="en">All contents</definition>
        </documentation>
      </annotation>
    </enumeration>
    <enumeration value="1">
      <annotation>
        <documentation>
          <label xml:lang="en">Level 1</label>
          <definition xml:lang="en">Level 1 contents</definition>
        </documentation>
```

```
      </annotation>
    </enumeration>
    <enumeration value="2">
      <annotation>
        <documentation>
          <label xml:lang="en">Level 2</label>
          <definition xml:lang="en">Up to level 2</definition>
        </documentation>
      </annotation>
    </enumeration>
    <enumeration value="3">
      <annotation>
        <documentation>
          <label xml:lang="en">Level 3</label>
          <definition xml:lang="en">Up to level 3</definition>
        </documentation>
      </annotation>
    </enumeration>
    <enumeration value="4">
      <annotation>
        <documentation>
          <label xml:lang="en">Level 4</label>
          <definition xml:lang="en">Up to level 4</definition>
        </documentation>
      </annotation>
    </enumeration>
    <enumeration value="5">
      <annotation>
        <documentation>
          <label xml:lang="en">Level 5</label>
          <definition xml:lang="en">Up to level 5</definition>
        </documentation>
      </annotation>
    </enumeration>
  </restriction>
</simpleType>

<complexType name="tPVRProfile">
  <sequence>
    <annotation>
      <documentation>
        Unit of the StorageLimitInVolume element is the GigaOctet
      </documentation>
    </annotation>
    <element name="PVRPreference" type="tns:tPVRPreference"/>
    <element name="StorageLimitInTime" type="tns:tStorageLimitInTime"
      minOccurs="0"/>
    <element name="StorageLimitInVolume" type="tns:tStorageLimitInVolume"
      minOccurs="0"/>
    <element name="Extension" type="tns:tExtension" minOccurs="0"/>
    <any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </sequence>
</complexType>

<simpleType name="tPVRPreference" final="list restriction">
  <restriction base="unsignedByte">
    <minInclusive value="0"/>
    <maxInclusive value="1"/>
    <enumeration value="0">
      <annotation>
        <documentation>
```

```
        <label xml:lang="en">Network</label>
        <definition xml:lang="en">
          Recording is done in the network
        </definition>
      </documentation>
    </annotation>
  </enumeration>
  <enumeration value="1">
    <annotation>
      <documentation>
        <label xml:lang="en">User_Equipment</label>
        <definition xml:lang="en">
          Recording is done on the user equipment
        </definition>
      </documentation>
    </annotation>
  </enumeration>
 </restriction>
</simpleType>

<simpleType name="tNPVRStorageLimitInTime">
 <restriction base="duration">
  <minInclusive value="PT0H"/>
  <maxInclusive value="PT1000000000H"/>
 </restriction>
</simpleType>

<simpleType name="tNPVRStorageLimitInVolume">
 <restriction base="nonNegativeInteger"/>
</simpleType>

<complexType name="tGlobalSettings">
 <sequence>
  <element name="LanguagePreference" type="tns:tLanguage" minOccurs="0"/>
  <element name="UsersActionRecodable" type="tns:tUserActionRecordable"/>
  <element name="Extension" type="tns:tExtension" minOccurs="0"/>
  <any namespace="##other" processContents="lax" minOccurs="0"
   maxOccurs="unbounded"/>
 </sequence>
</complexType>

<simpleType name="tLanguage">
 <restriction base="string">
  <annotation>
   <documentation>
    <definition xml:lang="en">ISO 639-2 Language code</definition>
   </documentation>
  </annotation>
  <minLength value="3"/>
  <maxLength value="3"/>
 </restriction>
</simpleType>

<complexType name="tExtension">
 <sequence>
  <any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
 </sequence>
</complexType>

<simpleType name="tUserActionRecordable">
 <restriction base="boolean"/>
</simpleType>
```

```
</schema>
```

D.2 XML Schema for the UE Profile

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="org:oipf:iptv:UEProfile:2008-1"
  xmlns:tns="org:oipf:itpv:UEProfile:2008-1"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:tva="urn:tva:metadata:2007" elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <import namespace="urn:tva:metadata:2007"
    schemalocation="tva_metadata_3-1_v141.xsd" />
  <annotation>
    <documentation xml:lang="en">
      Defines the capabilities of the UE that is currently
      associated with the user
    </documentation>
  </annotation>

  <element name="UEInformation" type="tns:tUEProfile" />
  <complexType name="tUEProfile">
    <sequence>
      <element name="UserEquipmentID" type="tns:tUEID" />
      <element name="UserEquipmentClass" type="tns:tUserEquipmentClass" />
      <element name="Resolution" type="tns:tResolution"
        minOccurs="0" />
      <element name="SupportedEncodings" type="tns:tSupportedEncodings"
        minOccurs="0" maxOccurs="unbounded" />
      <element name="SupportedContentProtection"
        type="tns:tSupportedContentProtection"
        minOccurs="0" maxOccurs="unbounded" />
      <element name="IPEncapsulations" type="tns:tIPEncapsulations"
        minOccurs="0" maxOccurs="unbounded" />
      <any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
  </complexType>

  <simpleType name="tUEID" final="list restriction">
    <annotation>
      <documentation>
        <label xml:lang="en">User Equipment ID</label>
        <definition xml:lang="en">
          Unique Identifier for the UE(to be specified)
        </definition>
      </documentation>
    </annotation>
    <restriction base="string">
      <minLength value="0" />
      <maxLength value="16" />
    </restriction>
  </simpleType>

  <simpleType name="tUserEquipmentClass"
    final="list restriction">
    <annotation>
      <documentation>
        <label xml:lang="en">User Equipment class</label>
        <definition xml:lang="en">
```



```
      Specifies the type of UE
     </definition>
    </documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="OITF-TV" />
    <enumeration value="OITF-STB" />
  </restriction>
</simpleType>

<complexType name="tResolution">
  <attribute name="HorizontalSize" type="integer">
    <annotation>
      <documentation>
        horizontal size in pixels of the screen
      </documentation>
    </annotation>
  </attribute>
  <attribute name="VerticalSize" type="integer">
    <annotation>
      <documentation>
        vertical size in pixels of the screen
      </documentation>
    </annotation>
  </attribute>
  <attribute name="Rotate" type="boolean">
    <annotation>
      <documentation>
        set to TRUE if the screen can be rotated (horizontal
        becomes vertical)
      </documentation>
    </annotation>
  </attribute>
</complexType>
<complexType name="tSupportedContentProtection">
  <annotation>
    <documentation>
      <label xml:lang="en">Content Protection</label>
      <definition xml:lang="en">
        Specifies the supported content protection system
        (eg. "urn:dvb:casystemid:19188") with optionally the gateway
        (eg. "CI+" or "DTCP-IP") and supported protected formats
      </definition>
    </documentation>
  </annotation>
  <sequence>
    <element name="ProtectedFormat" type="tns:tProtectedFormat"
      minOccurs="0" maxOccurs="unbounded"/>
    <any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </sequence>
  <attribute name="ContentProtectionSystemID" type="string" use="required"/>
  <attribute name="CSPG" type="tns:tCSPG" use="optional"/>
  <anyAttribute namespace="##any" processContents="lax"/>
</complexType>
<simpleType name="tCSPG">
  <annotation>
    <documentation>
      <label xml:lang="en">CSPG type</label>
      <definition xml:lang="en">
        Specifies the type of CSPG
      </definition>
```



```
    </documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="OIPF-CI+"/>
    <enumeration value="OIPF-DTCP-IP"/>
  </restriction>
</simpleType>
<simpleType name="tProtectedFormat">
  <annotation>
    <documentation>
      <label xml:lang="en">Protected Format</label>
      <definition xml:lang="en">
        Specifies the supported Protected Format
      </definition>
    </documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="BBTS"/>
    <enumeration value="PF"/>
    <enumeration value="PDCF"/>
    <enumeration value="MPIMP"/>
    <enumeration value="DCF"/>
  </restriction>
</simpleType>
<complexType name="tSupportedEncodings">
  <annotation>
    <documentation>
      <label xml:lang="en">encodings</label>
      <definition xml:lang="en">
        Specifies the supported audio and video encodings
        (eg. MPEG2,H264 AC3, AAC etc)
      </definition>
    </documentation>
  </annotation>
  <sequence>
    <element name="AudioEncoding" type="tns:tAudioEncoding"
      minOccurs="0" maxOccurs="unbounded" />
    <element name="VideoEncoding" type="tns:tVideoEncoding"
      minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>

<complexType name="tAudioEncoding">
  <annotation>
    <documentation>
      <label xml:lang="en">Audio Encoding</label>
      <definition xml:lang="en">
        Specifies supported audio encoding Properties
      </definition>
    </documentation>
  </annotation>
  <sequence>
    <element name="Encoding" type="tva:ControlledTermType" />
    <any namespace="##other" processContents="lax"
      minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>

<complexType name="tVideoEncoding">
  <annotation>
    <documentation>
      <label xml:lang="en">Video Encoding</label>
```

```
      <definition xml:lang="en">
        Specifies supported video encoding properties
      </definition>
    </documentation>
  </annotation>
  <sequence>
    <element name="Encoding" type="tva:ControlledTermType" />
    <element name="SupportedFrameRate" type="tva:FrameRateType"
      minOccurs="0" maxOccurs="unbounded" />
    <any namespace="##other" processContents="lax"
      minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>

<simpleType name="tIPEncapsulations" final="list restriction">
  <annotation>
    <documentation>
      <label xml:lang="en">encapsulation</label>
      <definition xml:lang="en">
        Specifies the IP encapsulation that is supported on
        the device (UDP/RTP, UDP/M2TS, UDP/RTP/M2TS)
      </definition>
    </documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="UDP/RTP"/>
    <enumeration value="UDP/M2TS"/>
    <enumeration value="UDP/RTP/M2TS"/>
  </restriction>
</simpleType>

<complexType name="tExtension">
  <sequence>
    <any processContents="lax" minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>

</schema>
```

D.3 IPTV Subscription Profile Elements classification

In this section, IPTV Subscription Profile elements are classified according to the desired visibility to the user:

- User visible and manageable data
- User visible, but not user-manageable data
- Data neither visible nor manageable by the user (service parameters that remain in the network, etc.)

Elements may remain without classification (that would mean that it is not determined if the data should remain in the network or visible to the user.)

element="IPTVProfile"

attribute ="ProfileId"

D.3.1 User visible and manageable data

CoD Profile:

In complexType ="tCoDProfile"

Element ="ParentalControl" type="tParentalControlLevel

Global Settings:

In complexType ="tGlobalSettings"

Element="LanguagePreference" type=tLanguage

D.3.2 User visible, but not manageable data

The term UE (User Equipment) is equivelant to the Open IPTV Forum term OITF.

In complexType="tUEProfile"

Element ="UserEquipmentID" type="tUEID"

Element ="UECapabilities" type="tUECapabilities"

In complexType="tUECapabilities"

Element ="UserEquipmentClass" type="UserEquipmentClass"

N-PVR:

In complexType="tPVRProfile"

Element="PVRPreference" type ="tPVRPreference"

Element ="StorageLimitInTime" type="tNPVRStorageLimitInTime"

Element ="StorageLimitInVolume" type="tNPVRStorageLimitInVolume"

D.3.3 Data neither visible nor manageable by the user

BC Profile:

In complexType ="tBCProfile"

Element ="BCServicePackage" type="tBCServicePackage"

In complexType ="tBCServicePackage"

Element name="BCPackageId" type="tBCServicePackageID"

Element name="Description" type="tBCServicePackageDescription"

Element name="BCService" type="tBCService"

complexType ="tBCService"

Element name="BCServiceId" type="tBCServiceID"

Element name="QualityDefinition" type="tQualityDefinition"

Annex E Mapping attributes for Scheduled Content

E.1 Mapping SDP attributes from DVB SD&S information

IP SDP parameters for each media stream	Corresponding DVB SD&S element in [TS 102 034] section 5.2.6.2 tables 4, 5 and 8.
Scheduled Content stream	
Connection Data c=<network type> <address type> <connection address>	
<network type>	Not retrieved from SD&S
<address type>	Not retrieved from SD&S
<connection address>	IPMulticastAddress@Address
Media Announcements for content delivery m=<media> <port> <proto> <fmt >	
<media>	"video" (also present in SD&S)
<port>	IPMulticastAddress@Port
<proto>	“RTP/AVP” if IPMulticastAddress@Streaming=“rtp” or if IPMulticastAddress@Streaming is not present
	“MP2T/H2221/UDP” or “RAW/RAW/UDP” if IPMulticastAddress@Streaming=“udp”
<fmt>	When MPEG2-Transport Stream [MPEG2-TS] is used, <fmt> shall be “33” as specified in RFC 3551. When optional Timestamped-TS defined by [DLNA] is used, the RTP/AVP dynamic payload type shall be used and <encoding name> of "a=rtpmap" line shall be "vnd.dlna.mpeg-tts" as specified in [DLNA]. Example m=video 49232 RTP/AVP 98 a=rtpmap:98 vnd.dlna.mpeg-tts/27000000
Bandwidth b=AS:<bandwidth>	MaxBitrate (Note: The “MaxBitrate” attribute in SD&S is calculated according to the TIAS bandwidth modifier defined in RFC 3890, but expressed in kb/s. The OITF should do the necessary conversion to express the bandwidth in the SDP as b=AS:<bandwidth>.)
BCServiceId	TextualIdentifier@ServiceName“:”TextualIdentifier@DomainName Note that the TextualIdentifier@DomainName is an optional attribute; therefore when not present, it is copied from the OfferingBase@DomainName
BCPackageId	Package@Id
FEC stream Note that the multicast address and source address of the FEC stream can be the same as the Scheduled Content stream.	
Media Announcements for FEC delivery m=<media> <port> <proto> <fmt>	Note: the FEC delivery can only be associated to a RTP delivered content.
<media>	“application”, not retrieved from SD&S
<port>	IPMulticastAddress.FECBaseLayer@Port
<proto>	RTP/AVP
<fmt>	Dynamic payload type

a=rtpmap:<fmt> <encoding_name/clock_rate>	<encoding_name/clock_rate> referring to the DVB-IP AL-FEC Base layer and is equal to: “vnd.dvb.iptv.alfec-base/90000”
Connection Data at media level c=<network type> <address type> <connection address>	
<network type>	Not retrieved from SD&S
<address type>	Not retrieved from SD&S
<connection address>	IPMulticastAddress.FECBaseLayer@Address

E.2 Service Package SDP attributes

The format of the a=bc_service_package attribute is the following:

a= bc_service_package : <BCPackageId> [mult_list] [bc_tv_service_id_list]

where

<mult_list> ::= mult_list:<source_unit>{“|”<source_unit>}

<source_unit> ::= [src_list:”(“<src-list>"),"]<multicast_address>{(“,”|”-“)<multicast_address>}

<src-list> ::= <source_address>{(“,”|”-“)<source_address>}

<source_address> ::= <IP_address>

<multicast_address> ::= <IP_address>

<bc_tv_service_id_list>::=<BCServiceId> {“,”<BCServiceId>}

<BCServiceId> is the string defined above.

<BCPackageId> is the service package ID string defined above.

(BNF notation). As seen in this notation the multi_list parameter can contain one or more source_unit parameters with multicast addresses that can be separated with either “,” or “-“.

When they are separated with “-“ it means that it is a range of addresses. In addition there can OPTIONALLY be a list of source addresses within the source unit parameter which is applicable for all the multicast addresses within the source unit parameter.

Annex F <protocol> names

F.1 Definition of <protocol> names

Following table shows the names (labels) of <protocol> which SHALL be a combination of signalling protocols and media transport protocols on UNI.

Table 64: Definition of <protocol> names

Service	Network Type	Signalling protocol	Media Transport protocol	Name of <protocol>
Scheduled Content	Managed	SIP + IGMP	RTP	“sip-igmp-rtp-udp”
			direct-UDP	“sip-igmp-udp”
	Unmanaged	IGMP	RTP	“igmp-rtp-udp”
CoD streaming	Managed	SIP + RTSP	RTP	“sip-rtsp-rtp-udp”
			direct-UDP	“sip-rtsp-udp”
	Unmanaged	RTSP	RTP	“rtsp-rtp-udp”
		N/A	HTTP	“http-get”
CoD download	Both	N/A	HTTP	“http-get”

Annex G System Infrastructure

G.1 OITF Start up High-Level Procedures

G.1.1 OITF with Native HNI-IGI Support

Figure 21 shows the high-level procedural flow for OITF starts up i.e. up to the point where all OITF functions are available. The following is a description of the steps:

Step 1: The local device start up procedure (which is implementation dependent).

Step 2: The OITF SHALL discover the IG through a UPnP procedure (section 10.1.1.1, "Procedure for IG Discovery").

G.2 Step 3: **The OITF SHALL use the DHCP option 124/125 to query the DHCP server to obtain the SP Discovery entry point. (See section 12.1.1.1.3, "Option 124/125".) If the deployment includes an IG, the DHCP server SHOULD[1] be configured to return the IG address in the DHCP option 125, either as a FQDN or as an IP address. In other words, in such a deployment the IG acts as the SP Discovery entry point (see section G.3, "OITF Restart high level procedures for an IG integrating GW**

This section details how stale SIP state can be detected in an IG integrating the GW, i.e. IG-GW, when an OITF restarts. This procedure is valid for both native and non-native HNI-IGI interfaces.

Figure 25 depicts how the IG-GW is able to establish a mapping between the SIP state (SIP dialog, IMPU and IP address) and the network state (IP address and deviceID).

The ability of the IG-GW to detect stale SIP state upon restart is based on the fact that when an OITF restarts, it re-initiates the DHCP server discovery (sends a DHCPDISCOVER message) and IP address request (DHCPREQUEST) procedures. This is an indication that the OITF has re-started. This is depicted in Figure 26.

[1] The DHCP server MAY, e.g., return the FQDN or IP address of the SP Discovery FE in the network instead.

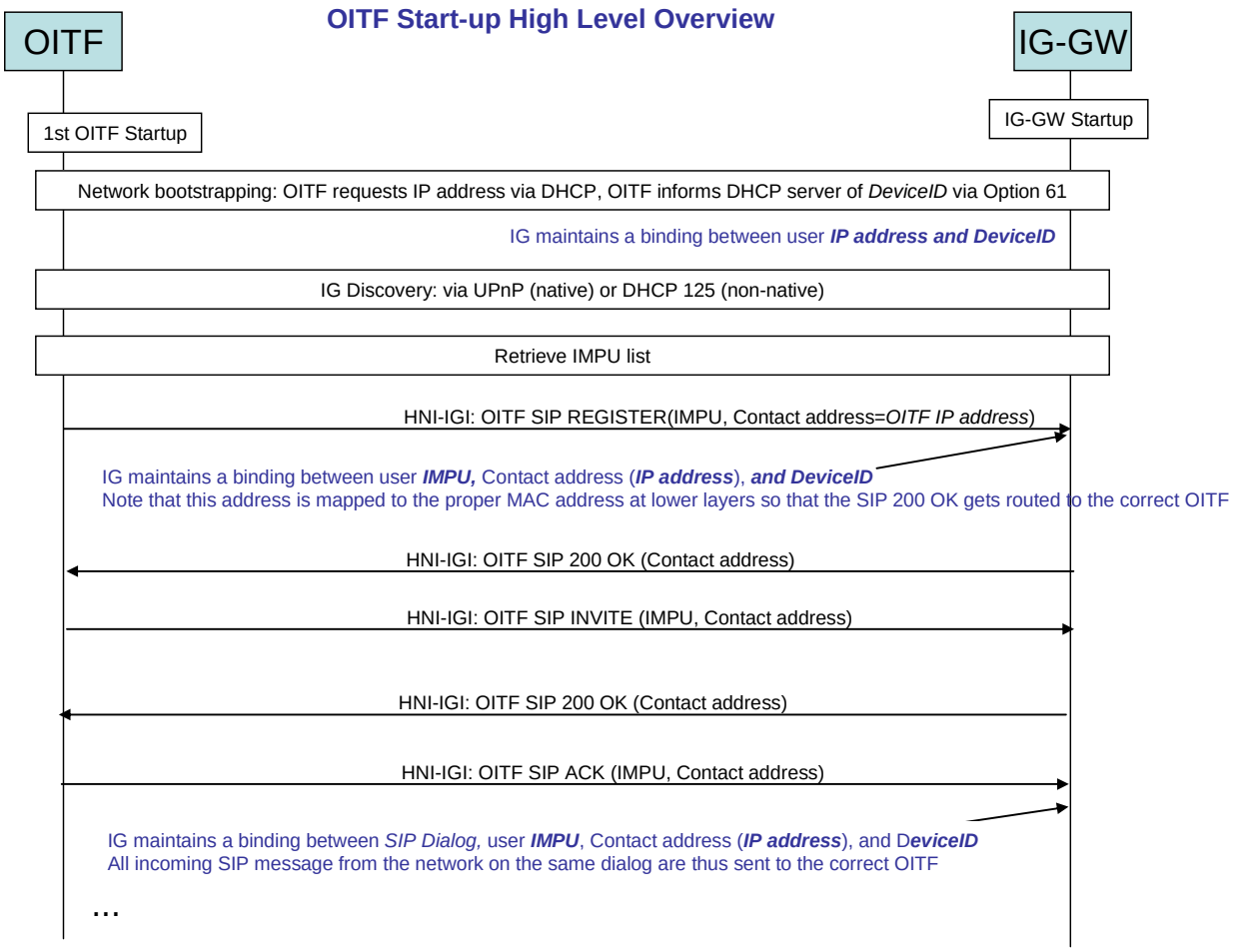


Figure 25: Overview OITF Start-up

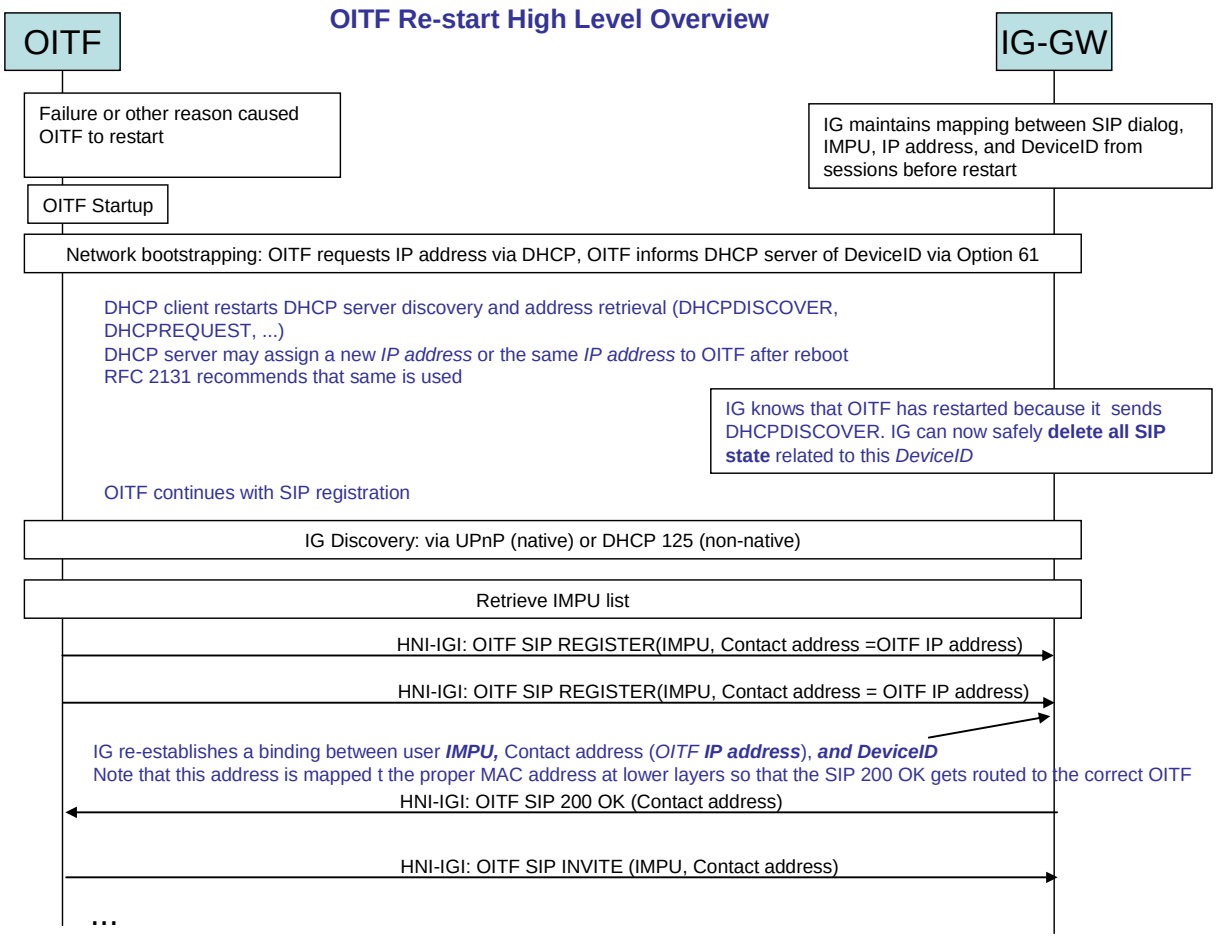


Figure 26: Overview OITF Restart

IG Startup and Shutdown procedures" for how an IG acts in this role).

Step 4: The OITF SHALL retrieve the list of subscription identities (IMPUs) (section 5.3.6.3, "User ID Retrieval for managed network services.")

Step 5: The OITF SHALL registers the user identity with the IG (section 5.3.6.1, "Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface.") An OITF without native support for HNI-IGI SHALL NOT perform this step *at this stage* (see step 9 below).

Step 6: The OITF SHALL perform GBA authentication (section 5.3.6.2.1, "Initial GBA registration.")

Step 7: The OITF SHALL perform service provider discovery (section 5.3.1, "Service Provider Discovery.") The service provider information MAY be returned directly by the IG.

Step 8: The OITF (or the DAE application, whichever applies) SHALL prompt the user to choose an SP. For any device, the timing and method of presentation as well as the relative positioning of the different SPs to the user is out of scope of the IPTV Solution specifications.

Step 9: The OITF (or the retrieved DAE application, whichever applies) SHALL perform service discovery (see section 5.3.2, "Service Discovery".)

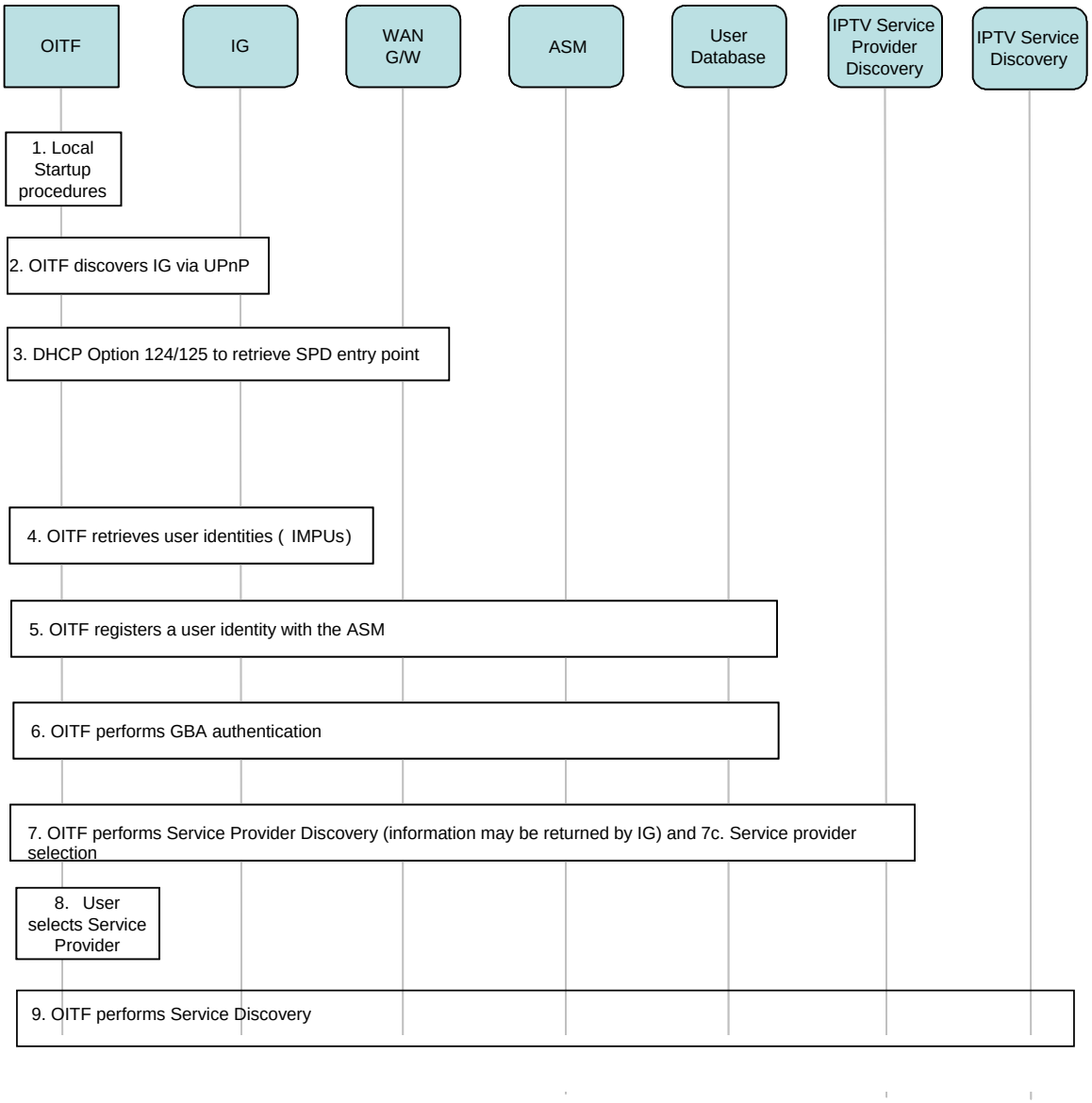


Figure 21: High level Start up procedural flow for an OITF with native HNI-IGI support

G.2.1 OITF with Non-Native HNI-IGI Support

Step 1: The local device start up procedure (which is implementation dependent).

G.3 Step 2: **The OITF SHALL use the DHCP option 124/125 to query the DHCP server to obtain the SP Discovery entry point. (See section 12.1.1.1.3, “Option 124/125”.) If the deployment includes an IG, the DHCP server SHOULD[2] be configured to return the IG address in the DHCP option 125, either as a FQDN or as an IP address. In other words, in such a deployment the IG acts as the**

[2] The DHCP server MAY, e.g., return the FQDN or IP address of the SP Discovery FE in the network instead.

SP Discovery entry point (see section G.4, “OITF Restart high level procedures for an IG integrating GW

This section details how stale SIP state can be detected in an IG integrating the GW, i.e. IG-GW, when an OITF restarts. This procedure is valid for both native and non-native HNI-IGI interfaces.

Figure 25 depicts how the IG-GW is able to establish a mapping between the SIP state (SIP dialog, IMPU and IP address) and the network state (IP address and deviceID).

The ability of the IG-GW to detect stale SIP state upon restart is based on the fact that when an OITF restarts, it re-initiates the DHCP server discovery (sends a DHCPDISCOVER message) and IP address request (DHCPREQUEST) procedures. This is an indication that the OITF has re-started. This is depicted in Figure 26.

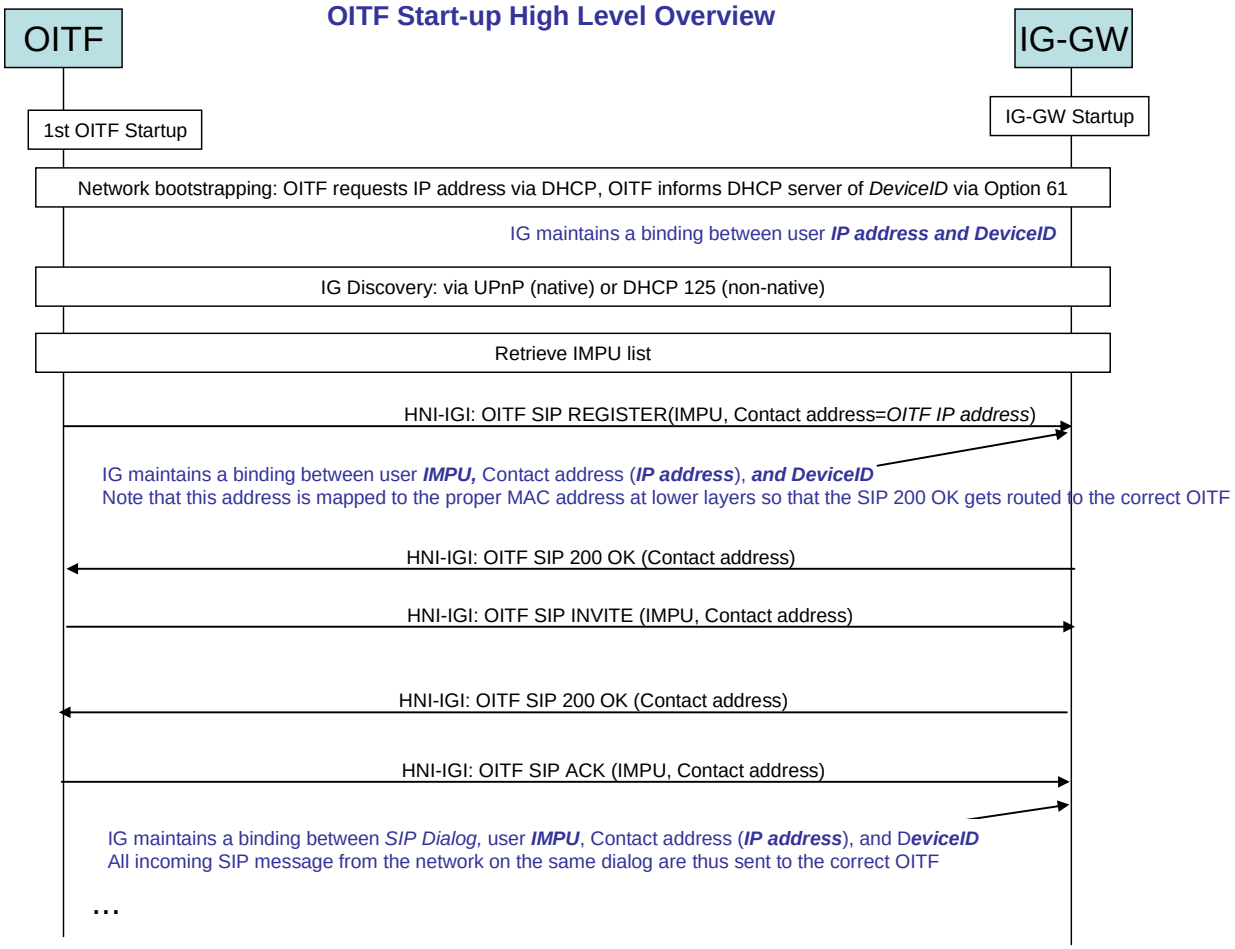


Figure 25: Overview OITF Start-up

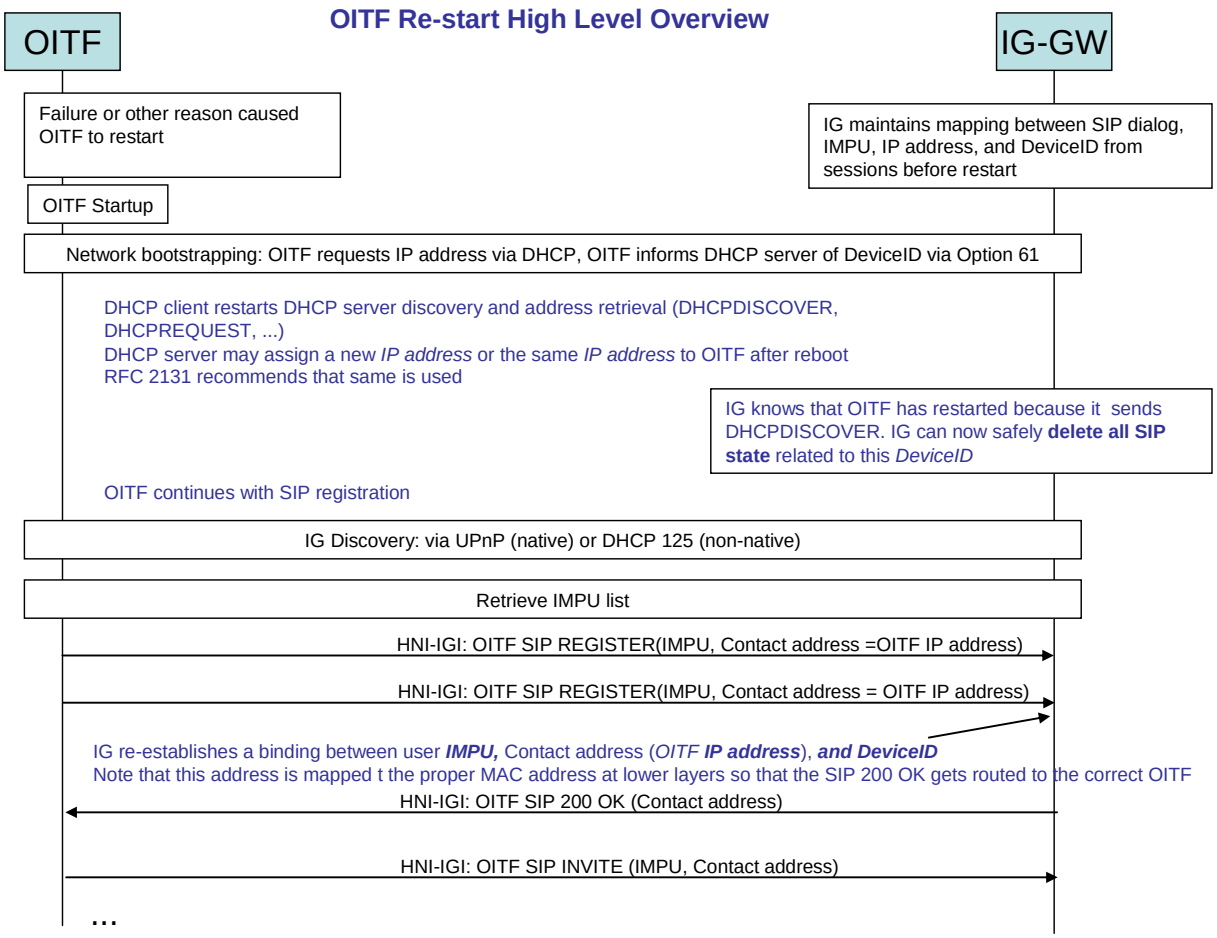


Figure 26: Overview OITF Restart

IG Startup and Shutdown procedures" for how an IG acts in this role).

Step 3: The OITF that has received the SP Discovery entry point via DHCP option 125 SHALL retrieve the Service Provider information by querying this entry point. (See section 5.3.1.2, "Protocol over UNIS-19 for the Unmanaged Model and Non-native HNI-IGI.") For any device, the time at which to trigger the query for Service Provider Discovery information is out of scope of the IPTV Solution specifications.

Step 4: The OITF (or the DAE application, whichever applies) SHALL prompt the user to choose an SP. For any device, the timing and method of presentation as well as the relative positioning of the different SPs to the user is out of scope of the IPTV Solution specifications.

Step 5: The OITF SHALL retrieve the list of user identities from the IG using the DAE application retrieved in step 4 (see section 5.3.6.3, "User ID Retrieval for managed network services".)

Step 6: The OITF SHALL register a user identity with the service platform provider, using a DAE application retrieved in step 4 (see section 5.3.6.3, "Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface".)

Step 7: The OITF (or the retrieved DAE application, whichever applies) SHALL perform service discovery (see section 5.3.2, "Service Discovery".)

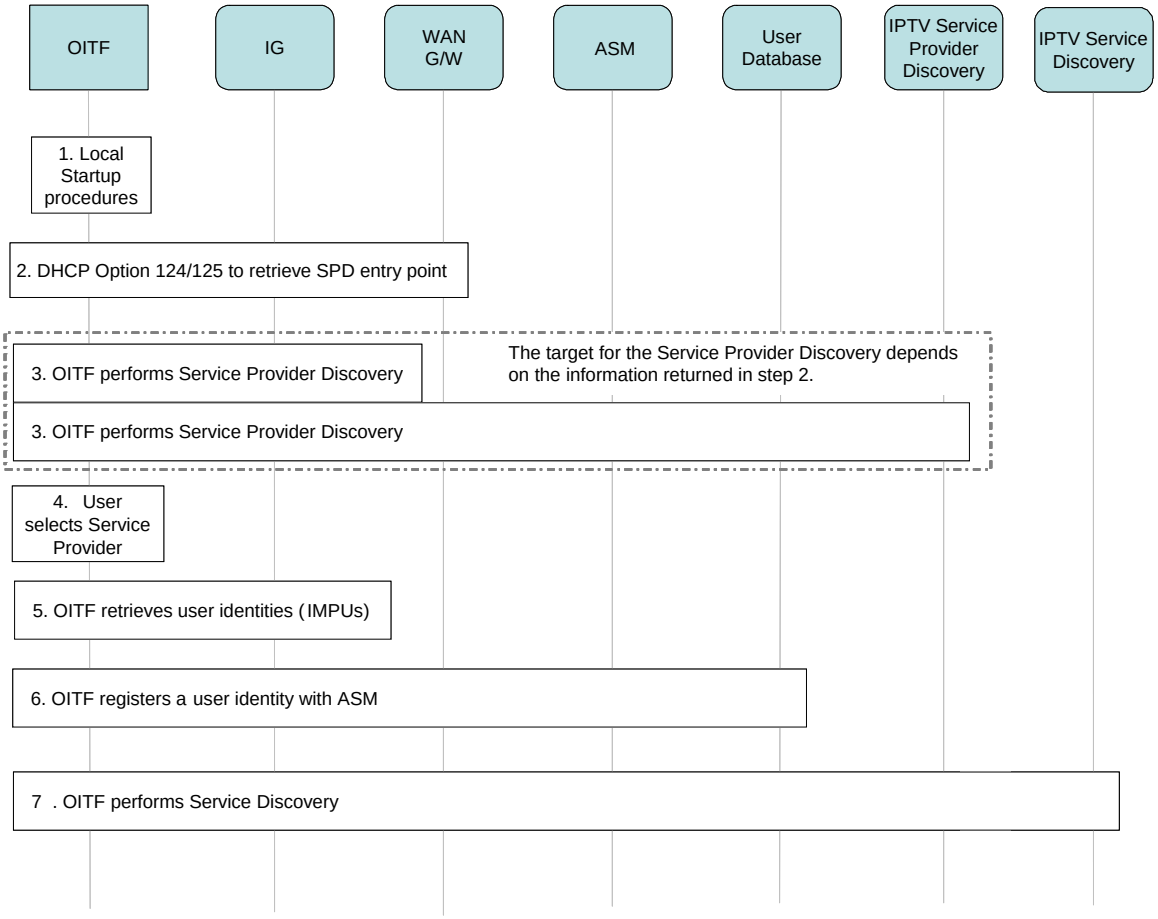


Figure 22: High-level start-up procedural flow for an OITF without native HNI-IGI support

G.3.1 Integrated OITF/IG with no HNI-IGI Support

Figure 23 shows the high-level procedural flow for an integrated OITF/IG device with no HNI-IGI support . The following is a description of the steps:

Step 1: The local device start up procedure (which is implementation dependent).

Step 2: The IG SHALL register the default user identity (section 5.3.6.1, “Procedure for User Registration and Authentication in the Managed Model on the HNI-IGI Interface.”).

Step 3: The IG SHALL perform GBA authentication (section 5.3.6.2.1, “Initial GBA registration.”).

Step 4a: The IG SHALL perform service provider discovery (section 5.3.1, “Service Provider Discovery.”).

Step 4b: The OITF (or the DAE application, whichever applies) SHALL prompt the user to choose an SP. For any device, the timing and method of presentation as well as the relative positioning of the different SPs to the user is out of scope of the IPTV Solution specifications.

Step 5: The OITF (or the retrieved DAE application, whichever applies) SHALL perform service discovery (see section 5.3.2, “Service Discovery”.)

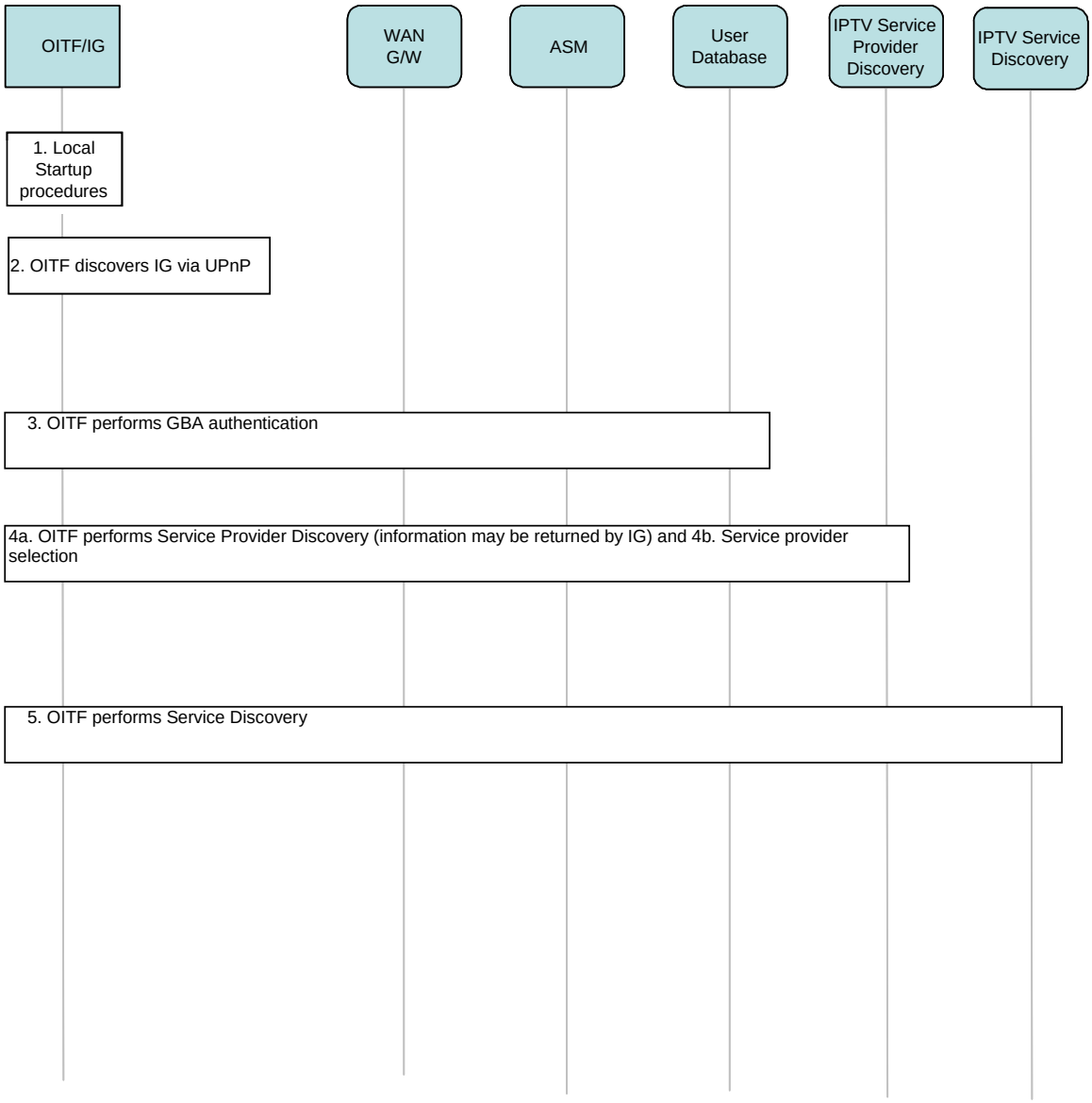


Figure 23: High-level start-up procedural flow for an integrated OITF/IG

G.4 High-Level Procedure for an OITF Graceful Shut Down for the Managed Model

Figure 24 shows a high-level procedural flow when an OITF is shut down, i.e. the OITF functionality is made completely inactive. The following steps are performed:

Step 1: The OITF gracefully terminates any ongoing IPTV session. (See appropriate IPTV Service Termination sections)

Step 2: The OITF de-registers the logged-in users (See section 5.3.6.1.2, “User De-registration.”)

Step 3: The IG terminates all activities for communication services for the de-registered identity that are associated with the OITF contact point (IP address) (See section 6.4, “Protocols for Communication Services.”)

Step 4: The IG de-registers the logged in user from the network. (See section 6.3.2.2, “Procedure for User Registration and Authentication in a Managed Model on UNIS-8.”) If this is the last OITF to shut down and if this is a default identity, then the IG must deregister the old contact point with the network and re-register the IG as the new contact point. If this is a user identity being deregistered, then the IG must deregister this identity and register a default identity with the IG as the contact point.

Step 5: The OITF performs local shut down procedure.

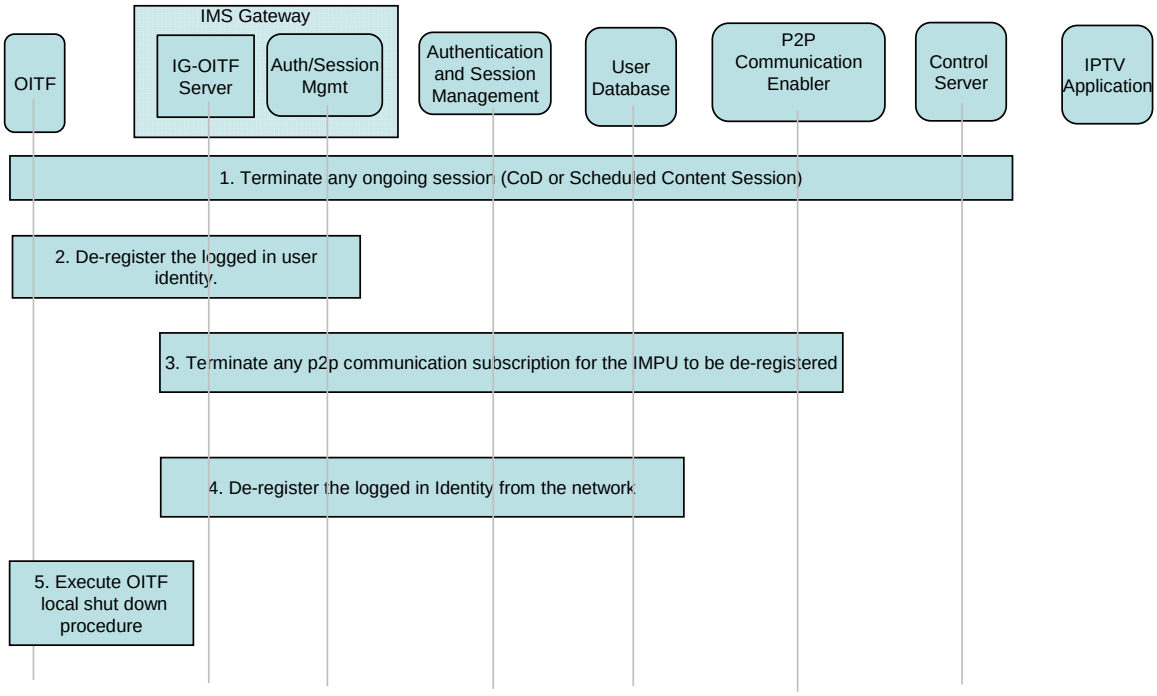


Figure 24: High level Shut-down procedural flow for an OITF

G.5 OITF Restart high level procedures for an IG integrating GW

This section details how stale SIP state can be detected in an IG integrating the GW, i.e. IG-GW, when an OITF restarts. This procedure is valid for both native and non-native HNI-IGI interfaces.

Figure 25 depicts how the IG-GW is able to establish a mapping between the SIP state (SIP dialog, IMPU and IP address) and the network state (IP address and deviceID).

The ability of the IG-GW to detect stale SIP state upon restart is based on the fact that when an OITF restarts, it re-initiates the DHCP server discovery (sends a DHCPDISCOVER message) and IP address request (DHCPREQUEST) procedures. This is an indication that the OITF has re-started. This is depicted in Figure 26.

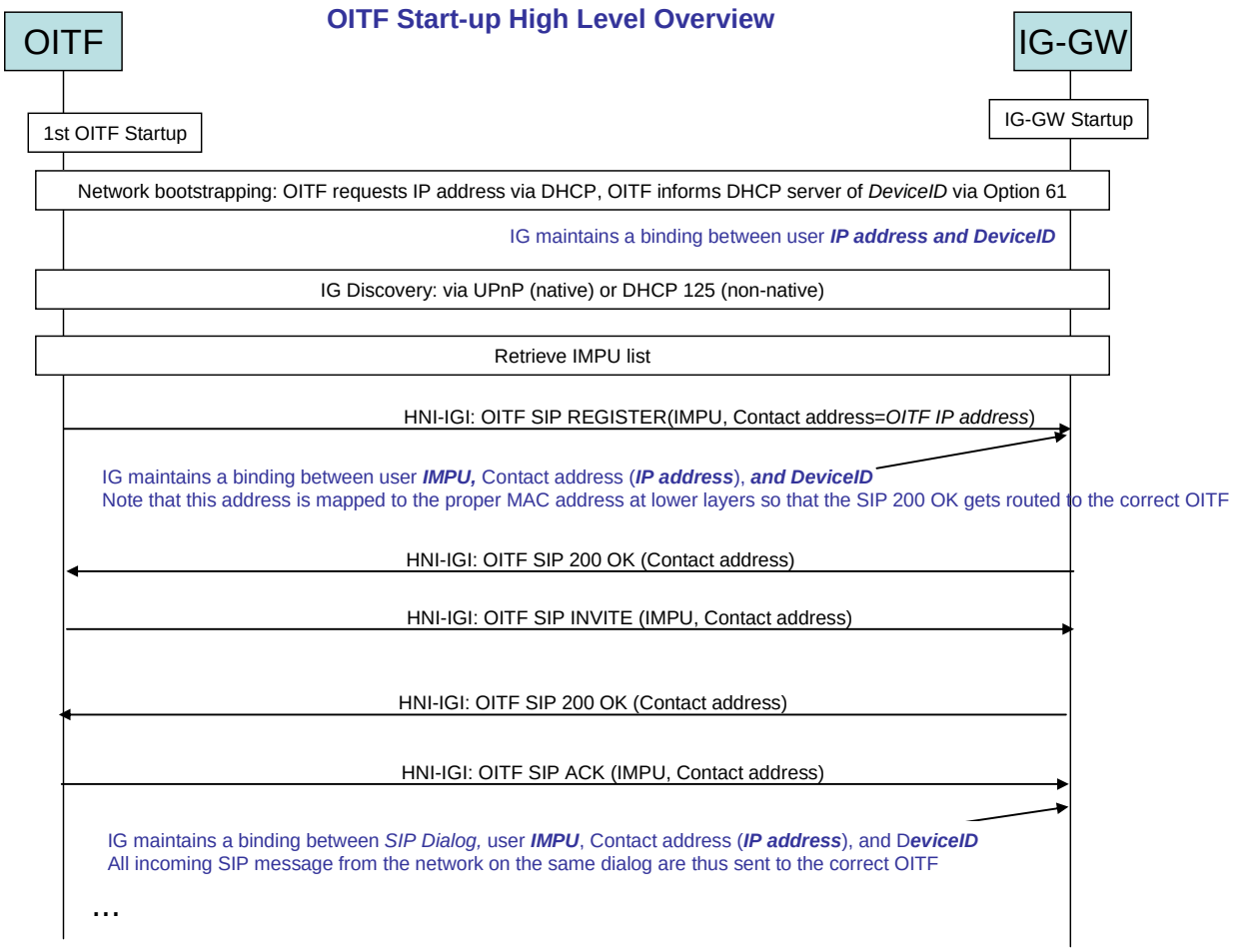


Figure 25: Overview OITF Start-up

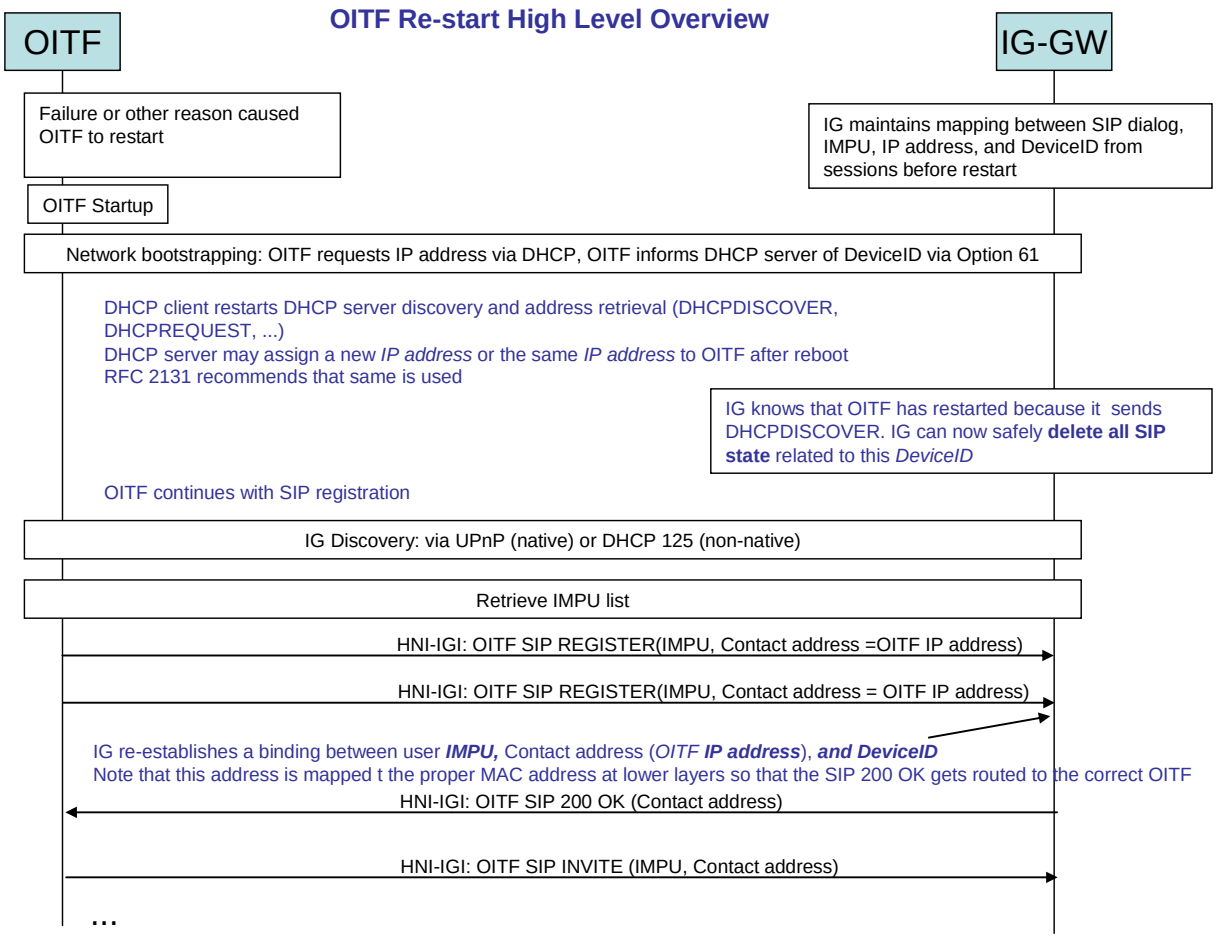


Figure 26: Overview OITF Restart

G.6 IG Startup and Shutdown procedures

G.6.1 IG Startup procedures

Step 1: IG power up initialization procedures (this is implementation dependent).

Step 2: The IG SHALL retrieve or receive user identities associated with the IMS subscription and other configuration information from the network (see section 5.3.5.1.3 Configuration of the IG via Configuration File)

Step 3: The IG SHALL register the default identity associated with the IMS subscription with the service platform (IMS) provider (see user registration section 6.3.2)

Step 4: The IG SHALL perform the SP Discovery procedure (see section 6.3.1.1 for details). The IG SHALL store the SP Discovery information in the format (see [META]) it was received.

At this point, the IG has completed its startup procedures and is ready to accept requests from the OITF and/or network entities.

The IG SHALL drop any messages received from the network related to the default user until such time that it detects that a default user has registered at an OITF.

G.6.2 IG Shutdown procedures

Step 1: The IG terminates all activities for communication services.

Step 2: The IG shall terminate all other SIP sessions.

Step 3: IG SHALL de-register all users bound to OITFs.

Step 4: The IG SHALL de-register any IG-initiated registration (if applicable).

Step 5: The IG performs its local shutdown procedure (implementation dependent)

G.7 WAN Gateway Functions

The WAN Gateway SHALL support multiple in-home devices for the consumer network, which SHALL be able to join the same multicast streams.

It SHOULD support IGMP snooping on all LAN side interfaces and forward inbound multicast packets to those physical interfaces which are connected to devices that have joined the specific multicast group.

The WAN Gateway SHALL support full IGMP v3 (RFC 3376).

It SHALL implement an IGMP proxy mechanism (RFC 4605).

G.8 NAT Traversal

The reason why IPTV will not function by default behind a NAT is that many of the communication parameters in SIP and in RTSP are transported within the SDP message; these parameters include the IP and port numbers used for signalling and media. A device behind a NAT does not know how it will be seen from the Network domain; it only knows its own IP address and the ports on the server where the application runs.

Once communication with a server starts, the NAT device translates the private IP and port combination of the device connected on the private NAT interface to a temporary mapping of a public IP and port on the interface connected to the public network.

When the Consumer Network uses a private IP addressing schema (e.g. RFC 1918) and the NAT device is port and/or address restricted, Consumer Network devices that receive incoming signalling (such as session setup, notification message, etc…) SHALL implement a mechanism to maintain open and active the necessary pin holes on the NAT device.

G.8.1 NAT Traversal for SIP based services for the Managed Model

The two main NAT traversal scenarios are summarised below:

G.8.1.1 Hosted NAT for SIP over IPSec

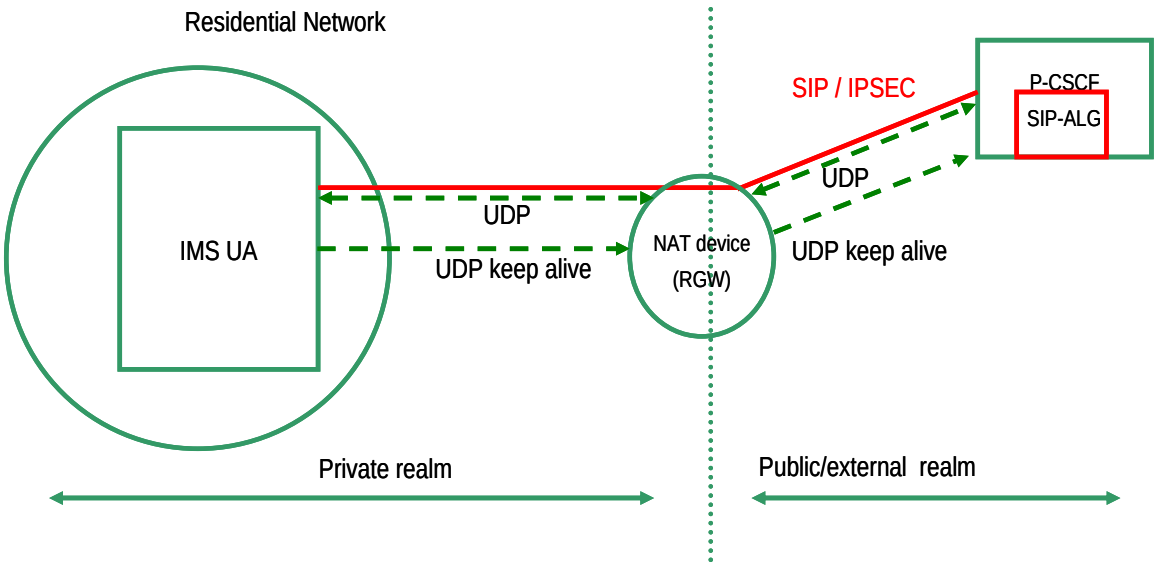
The NAT traversal solution defined for this scenario requires the following steps:

Step 1: Verify that the client (e.g. SIP UA) is behind a NAT device. In the IMS/3GPP scenario, this is achieved by using a plain text SIP message (the first SIP REGISTER). Note that within standard RFC IPSec the first step is performed directly within IKE (Internet Key Exchange), but within the IMS environment the authentication and key agreement phase is performed by using the AKA algorithm.

Step 2: The SIP UA establish the IPSec tunnel with the P-CSCF using the IETF IPSec NAT traversal solution that is based on UDP encapsulation;

Step 3: The UA maintains the pin holes open in the NAT device with UDP keep alive messages;

Step 4: All SIP message are sent over the IPSec tunnel (in both direction).



As there is a permanent communication path opened between the consumer device and the P-CSCF, it is always possible to send SIP messages between the entities involved in the communication (also when the SIP message request is originated from the network).

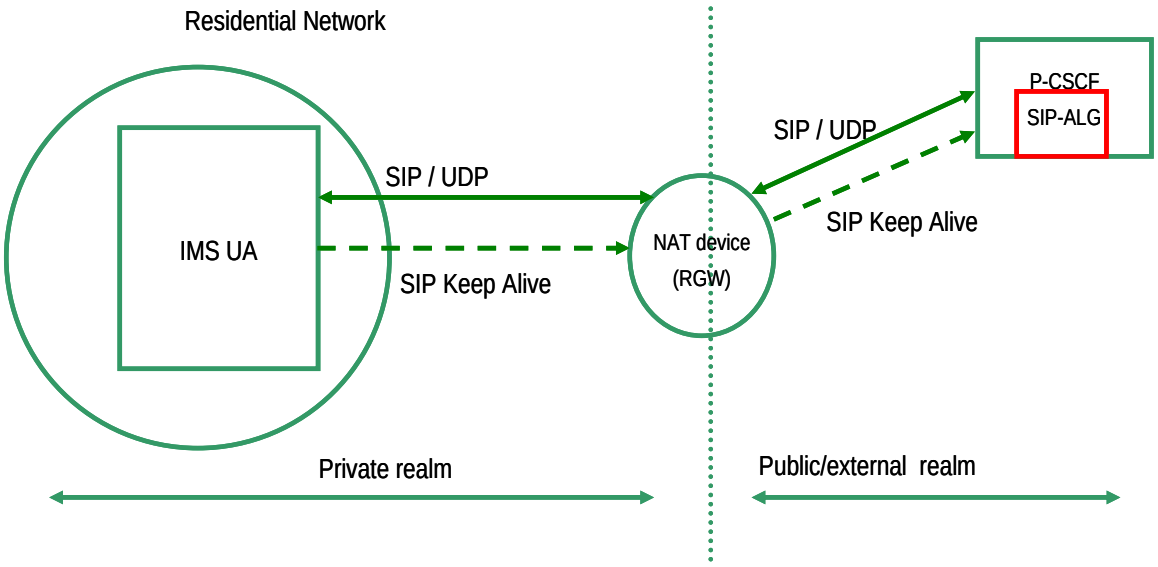
G.8.1.2 Hosted NAT for SIP plain text

The NAT traversal problem for SIP signalling can be solved by simply implementing keep alive messaging.

When it receives the first message, the P-CSCF can check the presence of a NAT device by comparing the address and port contained in the “Via” header to the actual IP and port combination in the received IP message.

Once registered with the SIP Registrar, the SIP UA must maintain the communication channel open by sending successive keep-alive packets before the binding expires in the NAT device.

The keep alive messages CAN either be SIP REGISTER or SIP OPTION sent by the client device.



As there is a permanent communication path opened between the consumer device and the P-CSCF, it is always possible to send SIP messages between the entities involved in the communication (also when the SIP message request is originated from the network).

It is required to use symmetric signalling, this means that the proxy must send and receive data on the same port number.

G.8.2 NAT Traversal and keep-alive messages for CoD

The NAT traversal problem for the media parameters transported within the SDP signalling can be solved by implementing a symmetric-RTP mechanism, as per RFC 4961 **[RFC4961]**:

- When the OITF activates a CoD service it SHALL start to send keep-alive messages that consist of empty RTP packets with a payload type of 20 to the appropriate destination address and port, which depend on the scenario of the deployment and the model.

In addition to the NAT traversal problem, there is a NAT binding keep-alive problem. In general, it is not possible to determine or modify a retail NAT's binding lifetime. Therefore, in order to keep the NAT bindings open, it is necessary to send keep-alives frequently. The required frequency of the keep-alive messages is governed by a keep-alive timer. The value of the keep-alive timer SHALL be a random number between 24 and 29 seconds, if not configured. This timer MAY be configurable as described in the RFC 4787 **[RFC4787]**. The empty RTP packet with a payload type of 20 is defined in TS 24.229 and endorsed in **[TS124503]**.

These empty RTP packets with payload type 20 fulfill the following functions:

1. The packets are used by the network for the discovery of the public client IP address and port (actually, the address and port of the WAN gateway) to use for the delivery of the RTP stream, and

2. The packets are also used to keep the necessary pin holes on the NAT device open and active for the duration of the incoming RTP streaming.

This solution applies for all cases with the following difference:

a) Managed Model: IG and WAN GW in different physical devices

   The functional entity that changes the destination address to the address and port discovered by the keep alive messages is the BGF (this is a component of the Transport Process Function defined in ES 282 003 **[ES282003]**), under the control of the P-CSCF (this is a component of the ASM defined in **[ARCH]**).

b) Managed Model: IG and WAN GW in one physical device

   In this scenario the IG+WAN GW device SHALL catch and suppress the keep-alive messages.

c) Unmanaged Model

   The functional entity that changes the destination address to the address and port discovered by inspecting the keep-alive messages is the CDF, under the control of the CC. The keep-alive message, reaching the CDF, provides the client's IP address and port to use for the delivery of the RTP stream.

Section 7.1.1.1, "RTSP Profile for the unmanaged model over UNIS-11 and NPI-10", gives some recommendations for slecting a mechanism for keeping the RTSP session "alive." In order to minimize OITF uplink traffic, the NAT binding keep-alives SHOULD be re-used as RTSP session keep-alive messages whenever possible. Note that it might be necessary for the OITF to send both NAT binding keep-alive messages and RTSP session keep-alives, for example, when the server cannot bind the RTP and RTSP sessions.

If the transport format of MPEG-2 TS encapsulated in UDP (direct UDP) is used, keep-alive messages with the following format SHALL be sent in order to keep the NAT bindings open: a UDP packet with body filled with bytes of value 20. The sender and destination IP/port settings follow the same rules as for RTP keep-alive messages.

Annex H gives more detail and describes the informational flow for these cases.

Annex H System Infrastructure Mechanisms (informative)

H.1 NAT-T Informational flows for Managed IPTV Services

The WAN Gateway itself can perform simple Network Address Translation (NAT) at the network and transport layer, but it is not able to modify the addresses embedded in the encapsulated signalling message. In order for the SIP services to work with NAT in this specification there are two possible alternatives that take into account the different deployment scenario defined in the architecture document **[ARCH]**:

1. If the WAN Gateway and the IG are deployed together in a physical device the NAT-T can be solved with an embedded SIP application-level gateway (SIP-ALG). In this scenario the SIP signalling is generated from this device using the public address and the control of the incoming media streams can be performed internally by the device.

2. For other deployment scenarios, the NAT-T can be solved in the network with a SIP application-level gateway (SIP-ALG) in the P-CSCF that coordinate the work of the BGF; this solution is commonly defined Hosted-NAT

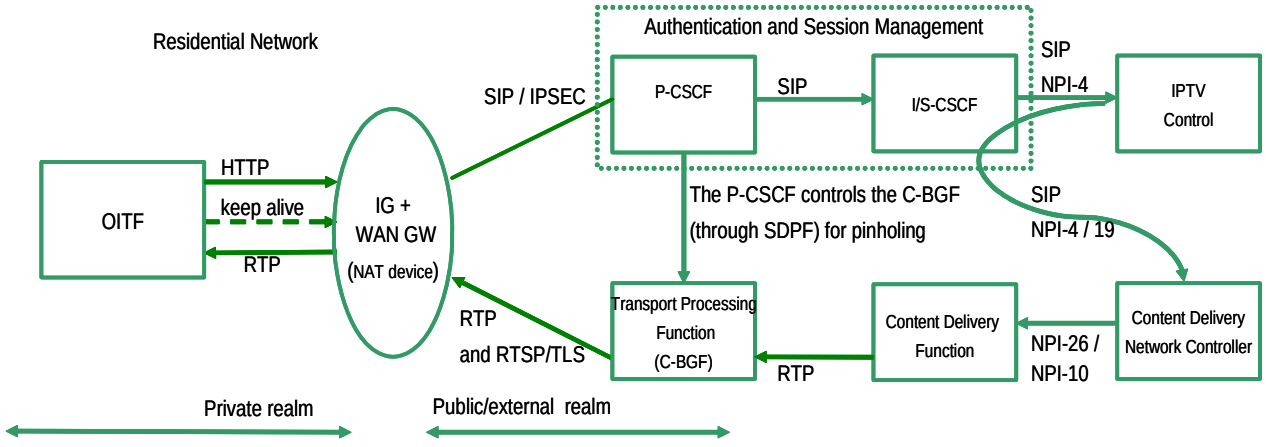
   The main advantages for the Hosted-NAT architectures are:

   - Minimal impact on the user device and the WAN gateway;
   - Security protocols supported (e.g. IPSec)
   - Main components already defined by TISPAN and 3GPP

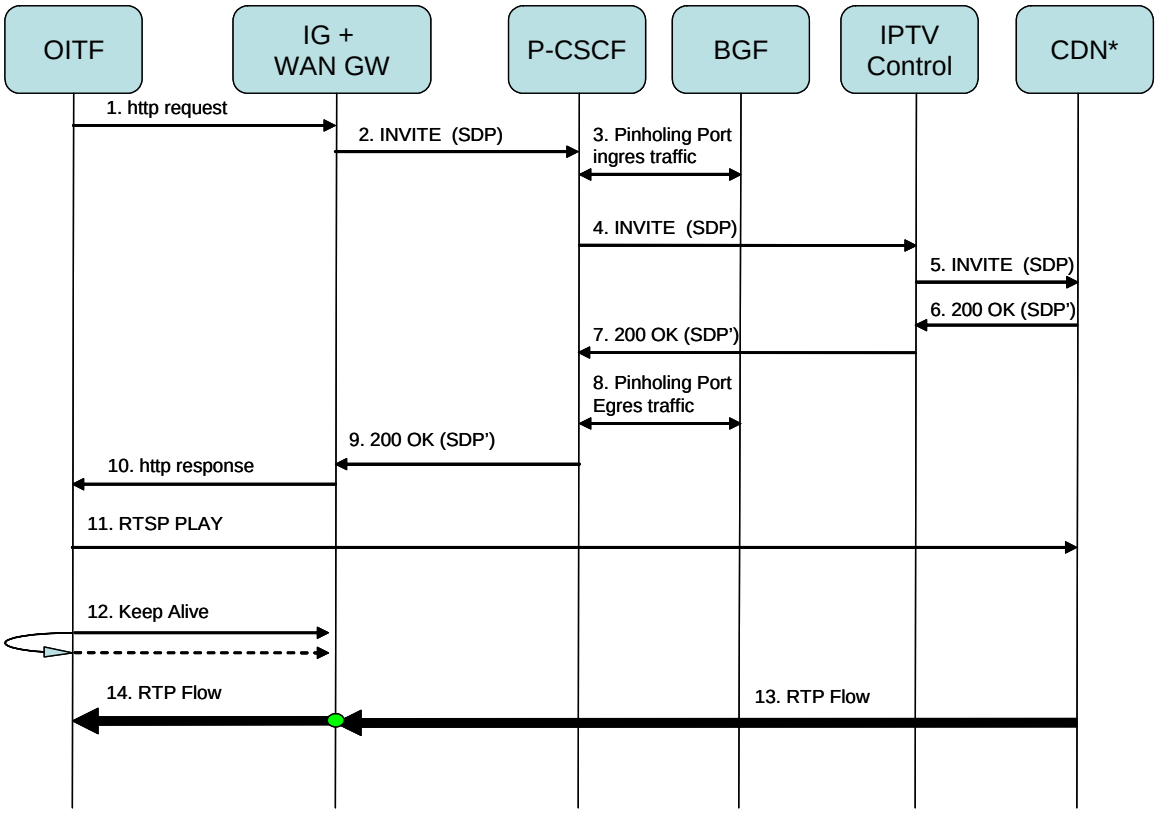
Since the Hosted-NAT solution is already used for other IMS services (e.g. voice call), it can be reused for the managed IPTV scenario in a general deployment option.

H.1.1 IG and WAN GW in one physical device

This section defines the NAT Traversal mechanism when the IG and WAN Gateway are deployed in the same device. This is considered to be a common scenario for managed networks. An embedded NAT-T solution and implemented internally in this device is considered an efficient mechanism.



The following informational flow describes the interaction between the functional entities defined by Open IPTV Forum for Content On-Demand services in this scenario, for simplicity the S-CSCF is not shown:



Step 1: The OITF sends a HTTP request for the desired CoD service to the IG (collocated with the WAN Gateway).

Step 2: The IG translates the request to a SIP INVITE with appropriate SDP description of the media request. The private address of the OITF is replaced with its public address.

Step 3: The P-CSCF (on Gq') requests to the BGF to open the pin hole for the ingress RTSP and eventually RTP media streams.

Step 4: The INVITE is forwarded to the IPTV Control.

Step 5: The INVITE is forwarded to the Cluster Delivery Network Controller (and Cluster Controller).

Step 6-7: The 200 OK message is sent back to the P-CSCF.

Step 8: The P-CSCF on Gq' updates the allocation on the BGF for the RTSP and RTP media streams (egress traffic).

Step 9: The 200 OK message is sent back to the IG+WAN GW.

Step 10: The information carried with the 200 OK is sent to the OITF (inside the HTTP replay message).

Step 11: The OITF send a RTSP PLAY command to receive the media stream to the Content Delivery Function

Step 12: The OITF starts sending keep alive messages that consist of empty RTP packet with a payload type of 20 to the destination address and port contained in the 200 OK.

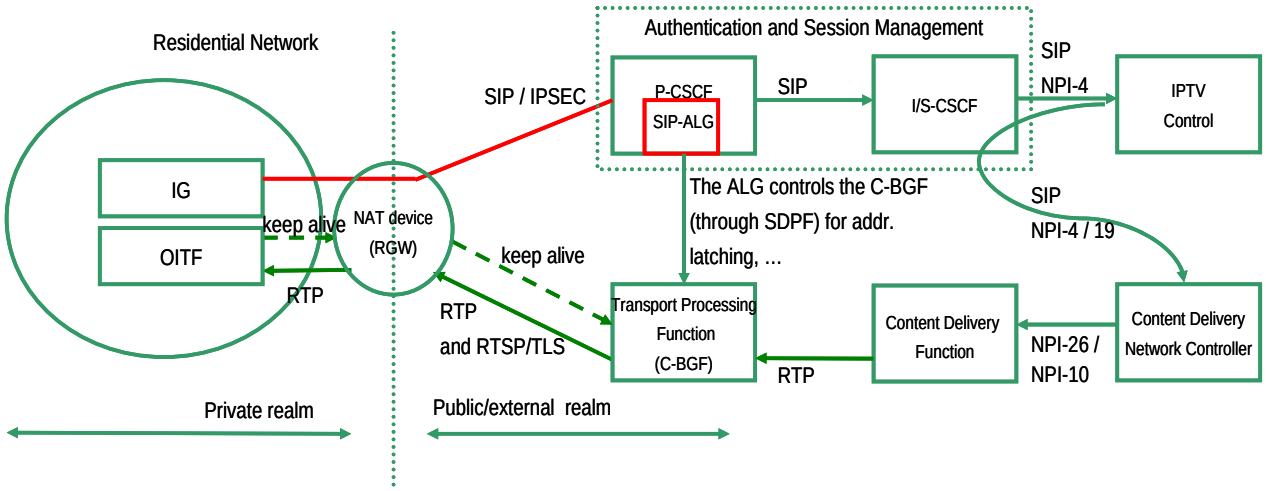
NOTE: In this scenario the IG+WAN GW device shall catch and suppress the keep alive messages.

Step 13: The CDN starts to send the media stream to the IG+WAN GW (by using the IP/Port received in the SDP packet as modified by the IG in step 2);

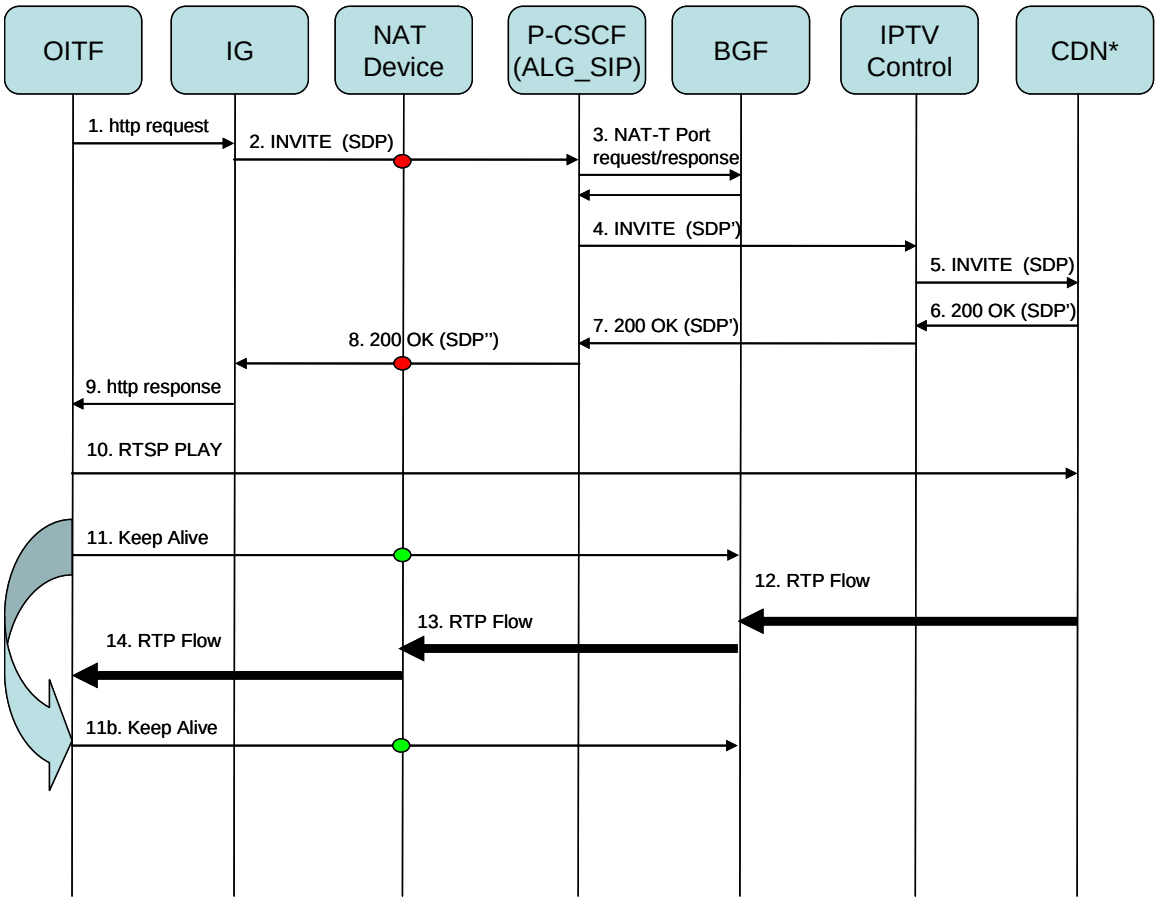
Step 14: The NAT device delivers the stream to the OITF by using its internal NAT table;

H.1.2 IG and WAN GW in different physical devices

In this scenario, the WAN GW is the NAT device and the solution is based on the 3GPP TS 33.203 IMS access NAT-T model.



It is required to maintain a permanent communication path opened between the IG and the P-CSCF. This can be achieved using the hosted NAT solution described in G.4.1. The following informational flow describes the interaction between the functional entities defined by Open IPTV Forum for Content on Demand services and explains the need for the RTP keep-alive messages; for simplicity the S-CSCF is not shown:



Step 1: The OITF sends a HTTP request for a CoD service to the IG.

Step 2: The IG translates the request to a SIP INVITE with appropriate SDP description of the media requested.

Step 3: The P-CSCF over the Gq' interface requests the allocation of address ports on the BGF for the RTP media streams; this information is also used to update the IP and port address in the SDP message that describes the RTP stream.

Step 4: The INVITE with SDP updated is sent to the IPTV Control FE.

Step 5: The INVITE is forwarded to the Cluster Delivery Network Controller (and the Cluster Controller).

Step 6-7: The 200 OK message is sent back to the P-CSCF.

Step 8: The SDP in the 200 OK answer is updated with the BGF address and port reserved for this media stream in the step 3.

Step 9: The information from the 200 OK is sent to the OITF.

Step 10: The OITF sends a RTSP PLAY command to receive the media stream from the Content Delivery Function.

Step 11: The OITF starts sending keep alive messages that consists of empty RTP packet with a payload type of 20 to the destination address and port contained in the 200 OK.

Step 12: The CDN starts to send the media stream to the BGF (by using the IP/Port received in the SDP packet).

Step 13: The BGF changes the destination address to the address and port discovered by the keep alive messages.

Step 14: The NAT device delivers the stream to the OITF by using its internal NAT table.

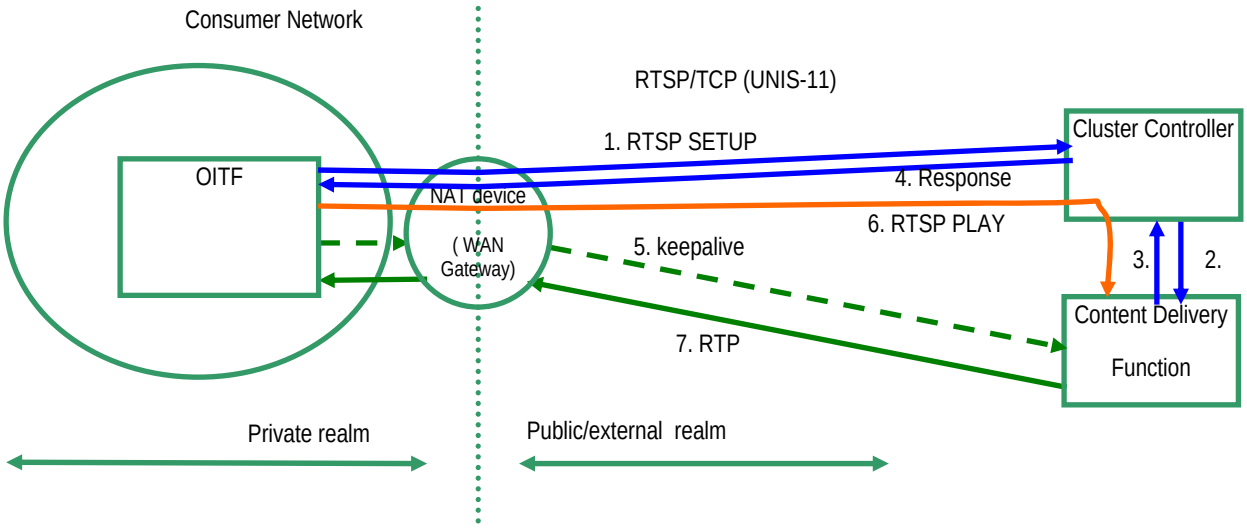
Note: The transport for RTSP is TCP only with either persistent or transient connection.

H.2 NAT Traversal for the Unmanaged Model

H.2.1 Symmetric-RTP for Unmanaged Model

In a similarly way to the managed network case, the Symmetric-RTP mechanism can also enable the NAT Traversal of RTP stream if the CDF for CoD service supports the detection of the external port number to send RTP stream towards the OITF. This solution will work even if multiple NAT devices exist between the CDF and the OITF.

The following flow describes the main steps involved in the Symmetric-RTP mechanism:



Step 1: The OITF sends an RTSP SETUP request to the CC (Cluster Controller). The CC detects the client's external IP address by the source IP address in the IP header.

Step 2: The CC forwards the information to the CDF.

Step 3: The CDF returns the server IP address and port number of the RTP stream.

Step 4: The CC returns the response for SETUP request to OITF which contains the server IP address and port number of CDF.

Step 5: The OITF sends the keep-alive messages that consist of empty RTP packet with a payload type of 20 to the CDF. The keep-alive message punches a hole in the NAT device and then, reaching the CDF, provides the client's IP address and port to use for sending the RTP stream.

Step 6: The OITF issues an RTSP PLAY request.

Step 7: The RTP packets can now be delivered from the CDF to the OITF.

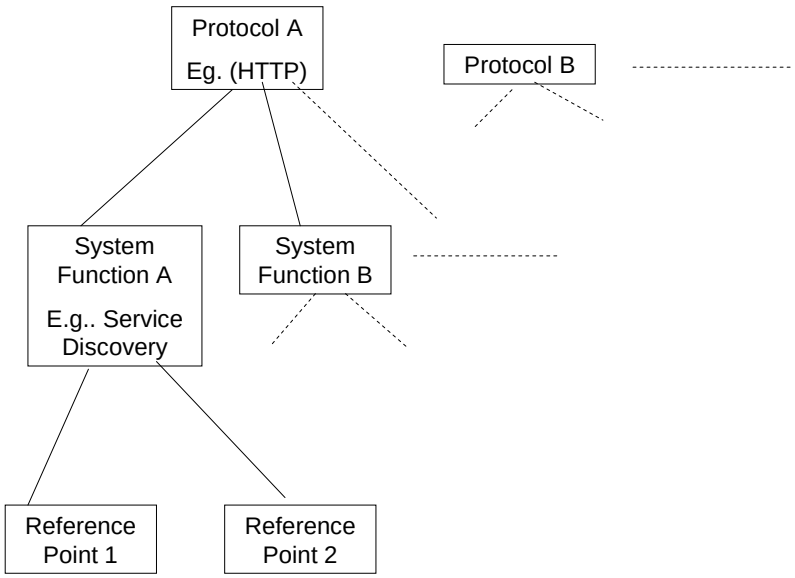
Annex I Presence XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="urn:oipf:service:oitfpresence:2008"
  xmlns:tns="urn:oipf:service:oitfpresence:2008"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:pdm="urn:ietf:params:xml:ns:pidf:data-model"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  version="0.1">
  <import namespace="urn:ietf:params:xml:ns:pidf:data-model"
    schemaLocation="data-model.xsd" />
  <!-- OMA extensions to PIDF tuple element for IPTV Presence services-->
  <import namespace="urn:oipf:service:oitfpresence:2008"
    schemaLocation="./iptv-IPTVProfile.xsd" />
  <!-- Import of the IPTV Profile elements -->
  <!-- list of definition of TISPAN element -->
  <simpleType name="tCurrentBCProgramID" final="list restriction">
    <restriction base="string">
      <minLength value="0" />
      <maxLength value="16" />
    </restriction>
  </simpleType>
  <simpleType name="tCurrentContentID" final="list restriction">
    <restriction base="string">
      <minLength value="0" />
      <maxLength value="16" />
    </restriction>
  </simpleType>
  <complexType name="tBCServicePresence">
    <sequence>
      <element name="CurrentBCServiceID" type="tBCServiceID" minOccurs="0" />
      <element name="CurrentBCProgramID"  type="tns:tCurrentBCProgramID"
        minOccurs="0" />
      <any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
  </complexType>
  <complexType name="tCoDServicePresence">
    <sequence>
      <element name="CurrentCoDContentID" type="tns:tCurrentContentID"
        minOccurs="0" />
      <any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
  </complexType>
  <complexType name="tNPVRServicePresence">
    <sequence>
      <element name="CurrentNPVRContentID" type="tns:tCurrentContentID"
        minOccurs="0" />
      <any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
  </complexType>
  <!-- end TISPAN basic element definition -->
  <simpleType name="hybridTechnologyType">
    <restriction base="string">
      <enumeration value="DVB-T" />
      <enumeration value="DVB-H" />
      <enumeration value="DVB-S" />
    </restriction>
  </simpleType>
  <complexType name="IPTVHybridType">
```

```
    <sequence>
      <element name="watchedBroadcast" type="tns:hybridContentType" />
      <element ref="pdm:deviceID" />
      <any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
    <attribute name="Technology" type="tns:hybridTechnologyType" />
  </complexType>
  <complexType name="hybridContentType">
    <sequence>
      <element name="currentChannel" type="string" />
      <element name="currentProgram" type="string" />
      <element name="serviceID" type="string" />
      <any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
  </complexType>
  <element name="IPTVHybridService">
    <complexType>
      <sequence maxOccurs="unbounded">
        <element name="IPTVHybrid" type="tns:IPTVHybridType" />
      </sequence>
    </complexType>
  </element>
</schema>
```

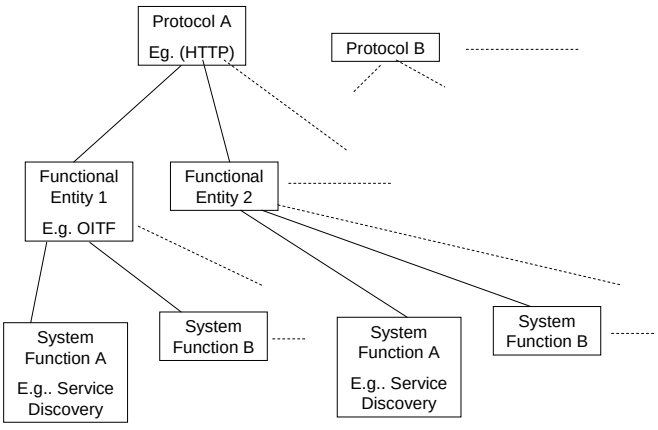
Annex J Protocol Procedure Section Structure (informative)

Each of the protocol sections of this document specifies the protocol procedures for a specific protocol (e.g. SIP, HTTP etc.). The sections have the following section and subsection structure.



This approach of structuring by protocol in the same way as the IMS IPTV Protocols specification developed in TISPAN should ensure that it is straightforward to investigate alignments with TISPAN. The structure chosen for this specification differs slightly from the TISPAN document structure, with the aim of helping the understanding of the end-to-end design, as it lends itself to the use of sequence charts to visualize the flow of a protocol through multiple components.

The actual TISPAN IMS IPTV Stage 3 specification structure is as follows:



Annex K OITF-specific TR-135 and TR-106 Remote Management Objects

A specific data model for the Remote Management of a retail OITF device has been defined. The data model has been obtained from TR-135 and TR-106 with a selection of a reduced set of parameters with the same semantics (with a few exceptions) and the same types.

K.1 OITF-specific TR-135 Remote Management Object

The following table, obtained from "Table 1/TR-135 – Parameter list for an STB CPE device" in TR-135 **[TR135]**, is the specific data model to manage an OITF in Release 1.

Table 65: Parameter list for an OITF using TR-135

Parameter	R/W	Description
.STBService.{i}.Capabilities.		**The overall capabilities of the OITF. This is a constant read-only object, meaning that only a firmware update will cause these values to be altered**
MaxActiveAVStreams	R	max no of simultaneous AV streams active
.STBService.{i}.Capabilities.PVR.		**PVR Capability**
MaxIOStreams	R	0 means no PVR function, 1 mean PVR function (0,1)
.STBService.{i}.Capabilities.AudioDecoder.		**Audio decoder capabilities**
AudioStandards	R	Comma-separated list of audio standards supported by this OITF
.STBService.{i}.Capabilities.VideoDecoder.		**Video decoder capabilities**
VideoStandards	R	Comma-separated list of video standards supported by this OITF
.STBService.{i}.Capabilities.VideoDecoder.MPEG4Part10.		**Object describing the set of supported profiles and levels for this OITF. It also describes the set of audio standards supported when MPEG4 Part 10 is used as the video standard.**
AudioStandards	R	Comma-separated list of supported Audio Standards supported by the Player when associated with MPEG4 Part 10 video. Each item is taken from the list defined by .Capabilities.AudioDecoder.AudioStandards
ProfileLevelNumberOfEntries	R	Number of instances of ProfileLevel
.STBService.{i}.Capabilities.VideoDecoder.MPEG4Part10.ProfileLevel.{i}.		**Table to describe the set of profiles and levels combinations supported by the OITF when MPEG4 Part 10 is used as video standard. Each entry in this table refers to a distinct combination of profile and level. The table MUST include a distinct entry for each supported combination of these parameters.**
Profile	R	Comma-separated list of supported MPEG4 Part 10 profiles. Each item is an enumeration of: “BASELINE” “MAIN” “EXTENDED” “HIGH”

		"HIGH 10" "HIGH 4:2:2" "HIGH 4:4:4"
Level	R	Comma-separated list of supported MPEG4 Part 10 Levels. Each item is an enumeration of: "1" "1b" "1.1" "1.2" "1.3" "2" "2.1" "2.2" "3" "3.1" "3.2" "4" "4.1" "4.2" "5" "5.1"
.STBService.{i}.Capabilities.DRM.		**This object describes the characteristics of the Conditional Access and/or Digital Rights Management of the OITF.**
DRMSystems	R	Comma-separated list of unique identifiers of OIPF supported Content Protection systems Each item is an enumeration: "urn:dvb:casystemid:19188" "OIPF-DTCP-IP" "OIPF-CI+" "urn:dvb:casystemid:456 OIPF-CI+" "urn:dvb:casystemid:12345 OIPF-DCTP-IP"
.STBService.{i}.Capabilities.ServiceMonitoring.		**This object describes the capabilities of the ServiceMonitoring object.**
MaxActiveMainStreams	R	Maximum number of AV Main streams for which the STB can simultaneously collect statistics.
MinSampleInterval	R	Minimum sample interval in seconds that the STB MUST be able to support.
MaxReportSamples	R	Maximum number of samples of each statistic that the STB is able to store and report.
.STBService.{i}.Capabilities.FrontEnd.		**Front-end capabilities.**
.STBService.{i}.Capabilities.FrontEnd.DVBT.		**Capabilities of the DVB-T receiver.**
MaxActiveDVBTStreams	R	Maximum number of simultaneous active AV streams supported by the DVB-T FrontEnd. (0, 1)
.STBService.{i}.Capabilities.FrontEnd.IP.		**IP Front-End capabilities.**
MaxDejitteringBufferSize	R	Describes the maximum de-jittering buffer size, in bytes, supported by the OITF.

.STBService.{i}.Components.		**Details of OITF logical or physical internal components.**
FrontEndNumberOfEntries	R	Number of FrontEnd instances.
AudioDecoderNumberOfEntries	R	Number of AudioDecoder instances.
VideoDecoderNumberOfEntries	R	Number of VideoDecoder instances.
DRMNumberOfEntries	R	Number of DRM instances.
.STBService.{i}.Components.FrontEnd.{i}.		
Name	R	
.STBService.{i}.Components.FrontEnd.{i}.DVBT.		**DVB-T front-end details.**
.STBService.{i}.Components.FrontEnd.{i}.DVBT.Modulation.		**DVB-T modulation details.**
SNR	R	This parameter is normally the Signal/Noise ratio in the carrier band, measured in dB. In the context of OITF, this parameter (0 to 10) gives a signal quality value from 0 (no signal), 1 (weak signal), 5 (medium signal) to 10 strong signal, when information is available from the chipset
.STBService.{i}.Components.FrontEnd.{i}.IP.		**DVB-T front-end details.**
InboundNumberOfEntries	R	Number of Inbound instances.
.STBService.{i}.Components.FrontEnd.{i}.IP.RTCP.		**Parameters related to RTCP receiver report generation**
Enable	W	Enables or disables RTCP receiver report generation. If the OITF does not implement RTCP, then the OITF SHALL send error code 9001, “Request denied (no reason specified)” when the server tries to enable RTCP feedback.
Status	R	The status of RTCP receiver report generation. Enumeration of: “Disabled” “Enabled” “Error” (OPTIONAL) The “Error” value MAY be used by the CPE to indicate a locally defined error condition.
.STBService.{i}.Components.FrontEnd.{i}.IP.Inbound.{i}.		**Inbound IP streams currently entering the OITF via this front-end.**
SourceAddress	R	IP address of the source of the current stream content.
SourcePort	R	TCP or UDP port number of the source of the current stream content, or 0 if the content is not being delivered via IP or if not applicable.
URI	R	RFC 3986 URI that indicates the current source (possibly including Multicast group and port, if relevant) of the stream content, or an empty string if the source is not known or cannot be represented as a URI.
.STBService.{i}.Components.AudioDecoder.{i}.		**Audio decoder instance table. It contains data representing the current status of the Audio decoder.**
Status	R	The status of this audio decoder.

		Enumeration of: "Disabled" "Enabled" "Error" (OPTIONAL) The "Error" value MAY be used by the CPE to indicate a locally defined error condition.
Name	R	Human-readable name associated with this audio decoder.
AudioStandard	R	Audio standard currently being processed by this audio decoder, or an empty string if no audio standard is currently being processed.
.STBService.{i}.Components.VideoDecoder.{i}.		**Video decoder instance table. It contains data representing the current status of the video decoder.**
Status	R	The status of this video decoder. Enumeration of: "Disabled" "Enabled" "Error" (OPTIONAL) The "Error" value MAY be used by the CPE to indicate a locally defined error condition.
Name	R	Human-readable name associated with this video decoder.
MPEG4Part10	R	Path name of the MPEG4 Part 10 profile and level object instance.
ContentAspectRatio	R	Indicates the native aspect ratio of the content available at this decoder. Enumeration of: "4:3" "16:9"
.STBService.{i}.Components.DRM.{i}.		**This object describes the characteristics of the Digital Rights Management**
Status	R	The status of this DRM system. Enumeration of: "Disabled" "Enabled" "Error" (OPTIONAL) The "Error" value MAY be used by the CPE to indicate a locally defined error condition.
Name	R	Indicates a unique identifier for this DRM system. This name MUST appear in the .Capabilities.DRM.DRMSystems list.
.STBService.{i}.AVStreams.		**AV Streams object. If more than one AV stream can be active at a time, it may contain several AVStream instances.**
ActiveAVStreams	R	Number of AV streams currently active
AVStreamNumberOfEntries	R	Number of AVStream instances.
.STBService.{i}.AVStreams.AVStream.{i}.		**Details of each AVStream. AV streams are created statically. Each AV stream corresponds to a valid {FrontEnd, Audio- Decoder, VideoDecoder} instance combination**
Status	R	The status of this AV stream. Enumeration of: "Disabled"

		"Enabled" "Error_PVRWriteFailure" "Error_PVRReadFailure" "Error" Unspecified error (OPTIONAL) An AV stream is disabled if any of the referenced objects are disabled. If an AV stream is disabled then the values of other AV stream parameters are not significant. The "Error" value MAY be used by the CPE to indicate a locally defined error condition.
Name	R	Human-readable name associated with this stream
FrontEnd	R	Path name of the input FrontEnd object instance associated with this AV stream.
AudioDecoder	R	Path name of the Audio Decoder object instance associated with this AV stream.
VideoDecoder	R	Path name of the Video Decoder object instance associated with this AV stream.
Inbound	R	Path name of the inbound IP stream object instance associated with the FrontEnd for this AV stream.
.STBService.{i}.ServiceMonitoring.		**Contains statistics relating to the QoS / QoE of Main AV streams. Note that OITF devices do not support the collection of statistics while in STANDBY mode.**
SampleEnable	W	Enables or disables collection of Sample statistics.
SampleState	R	Indicates availability of Sample statistics. Enumeration of: "Disabled" Collection is disabled "Enabled" Collection is enabled "Trigger" Collection is enabled and the ACS should now fetch the collected data The transition from Enabled -> Trigger -> Enabled MUST be instantaneous and so will result in only a single value change for notification purposes.
SampleInterval	W	The sample interval in seconds.
ReportSamples	W	The number of samples that the OITF can store and report for each statistic.
TimeReference	W	An absolute time reference in UTC to determine when sample intervals will complete.
ReportStartTime	R	The absolute time at which the sample interval for the first stored sample (for each statistic) started.
ReportEndTime	R	The absolute time at which the sample interval for the last stored sample (for each statistic) ended.
MainStreamNumberOfEntries	R	Number of MainStream instances.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.		**List of Main AV stream objects. Each instance is associated with a specified service type and will collect statistics only for the main stream that matches that service type.**
Enable	W	Enables or disables collection of Total and

		Sample statistics for this object instance.
Status	R	Total and Sample statistics collection status for this object instance. Enumeration of: “Disabled” “Enabled” “Error” (OPTIONAL) The “Error” value MAY be used by the CPE to indicate a locally defined error condition.
ServiceType	W	Service type associated with this main stream instance, or an empty string if this instance is disabled. ServiceType is taken from the list: “IPTV” “VoD” “IP” “CAB” “DTT” “SAT” “PVR”
AVStream	R	Path name of the Main AV stream object instance currently associated with this ServiceMonitoring main stream instance.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.		**Total statistics since this ServiceMonitoring main stream instance was last enabled or Total statistics were last reset.**
Reset	W	When set to true, resets Total statistics for this ServiceMonitoring main stream instance. Setting it to false has no effect. The value is not saved in device state and is always false when read.
ResetTime	R	Number of seconds since the Total statistics were last enabled or reset for this ServiceMonitoring main stream instance.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.DejitteringStats.		**Total de-jittering statistics for this ServiceMonitoring main stream instance.**
Overruns	R	Total number of times the receive jitter buffer has overrun for this AV stream.
Underruns	R	Total number of times the receive jitter buffer has underrun for this AV stream.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.RTPStats.		**Total RTP statistics for this ServiceMonitoring main stream instance.**
PacketsReceivedBeforeEC	R	Total number of RTP packets received for this AV stream. These statistics are collected before any EC, if available, is applied.
PacketsLostBeforeEC	R	Total number of RTP packets lost for this stream. These statistics are collected before any EC, if available, is applied.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.MPEG2TSStats.		**Total MPEG2-TS statistics for this ServiceMonitoring main stream instance.**
TSPacketsReceived	R	Total number of MPEG2-TS packets received for this AV stream.
PacketDiscontinuityCounter	R	Total number of MPEG2-TS Discontinuity

		errors that have been captured for this AV stream. This parameter accumulates all of the discontinuities observed for all currently monitored PIDs.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Total.VideoDecoderStats.		**Total video decoder application layer statistics for this ServiceMonitoring main stream instance.**
ILostFrames	R	The number of I frames that could not be reproduced by the OITF for this AV stream.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Sample.RTPStats.		**RTP Sample statistics for this Service-Monitoring main stream instance.**
SampleSeconds	R	Comma-separated list; each entry is the number of seconds during which RTP data was collected for this AV stream during the sample interval.
PacketsExpected	R	Comma-separated list; each entry is the total number of RTP packets expected for this AV stream during the sample interval
PacketsLostBeforeEC	R	Comma-separated list; each entry is the total number of RTP packets lost for this AV stream during the sample interval.
PacketsReceivedBeforeEC	R	Total number of RTP packets received for this AV stream. These statistics are collected before any EC, if available, is applied.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Sample.MPEG2TSStats.		**MPEG2-TS Sample statistics for this Service-Monitoring main stream instance.**
SampleSeconds	R	Comma-separated list; each entry is the number of seconds during which MPEG2-TS data was collected for this AV stream during the sample interval.
TSPacketsReceived	R	Comma-separated list; each entry is the total number of MPEG2-TS packets received for this AV stream during the sample interval.
PacketDiscontinuityCounter	R	Comma-separated list; each entry is the total number of MPEG2-TS Discontinuity errors that were captured for this AV stream during the sample interval.
.STBService.{i}.ServiceMonitoring.MainStream.{i}.Sample.DejitteringStats.		**De-jittering Sample statistics for this ServiceMonitoring main stream instance.**
SampleSeconds	R	Comma-separated list; each entry is the number of seconds during which de-jittering data was collected for this AV stream during the sample interval.
Overruns	R	Comma-separated list; each entry is the total number of times the receive jitter buffer has overrun for this AV stream during the sample interval.
Underruns	R	Comma-separated list; each entry is the total number of times the receive jitter buffer has underrun for this AV stream during the sample interval.

K.2 OITF-specific TR-106 Remote Management Object

The following table, obtained from "Table 3 – Common Object definitions for Device:1" in TR-106 Amendment 1, "Data Model Template for TR-069-Enabled Devices" **[TR106]**, is the specific data model to manage an OITF in Release 1.

Note that:

- For Device.DeviceInfo, the 3 parameters ManufacturerOUI, ProductClass and Serial Number have slightly different semantic meanings in the context of OIPF and are obtained from the deviceID identifier (refer to section 6.3.2.1, "User Identity Modelling"):
    - ManufacturerOUI = HEX(first 3 bytes of SHA-1(X))
    - ProductClass = "OIPF"
    - SerialNumber = HEX(remaining bytes, from 4th on, of SHA-1(X))

  where X = (MAC address as bytes) + (domain name in ASCII characters).

- All Device.LAN parameters are read-only

Table 66: Parameter list for an OITF using TR-106

Parameter	R/W	Description
Device.		
DeviceSummary	R	The DeviceSummary parameter is defined to provide an explicit summary of the top-level data model of the device, including version and profile information.
Device.DeviceInfo.		**General information about the device, including its identity and version information.**
Manufacturer	R	The manufacturer of the CPE (human readable string).
ManufacturerOUI	R	In the context of OIPF, this parameter is the hexadecimal value of the first 3 bytes of SHA-1(X)
ModelName	R	Model name of the CPE (human readable string).
Description	R	A full description of the CPE device (human readable string).
ProductClass	R	In the context of OIPF, this parameter is always "OIPF"
SerialNumber	R	In the context of OIPF, this parameter is the hexadecimal value of the remaining bytes (from 4th on) of SHA-1(X)
SoftwareVersion	R	A string identifying the software version currently installed in the CPE.
Device.ManagementServer.		**This object contains parameters relating to the CPE's association with an ACS.**
URL	W	URL for the CPE to connect to the ACS using the CPE WAN Management Protocol. This parameter MUST be in the form of a valid HTTP or HTTPS URL. The "host" portion of this URL is used by the CPE for validating the ACS certificate when using SSL or TLS. Note that on a factory reset of the CPE, the value of this parameter might be reset to its factory value. If an ACS modifies the value of this parameter, it SHOULD be prepared to accommodate the situation that the original value is restored as the result of a factory reset.
Username	W	Username used to authenticate the CPE when making a connection to the ACS using the CPE WAN Management Protocol. This username is used only for HTTP-based authentication of the CPE. Note that on a factory reset of the CPE, the value of this parameter might be reset to its factory value. If an ACS modifies the value of this parameter, it SHOULD be prepared to accommodate the situation that the

		original value is restored as theresult of a factory reset.
Password	W	Password used to authenticate the CPE when making a connection to the ACS using the CPE WAN Management Protocol. This password is used only for HTTP-based authentication of the CPE. When read, this parameter returns an empty string, regardless of the actual value. Note that on a factory reset of the CPE, the value of this parameter might be reset to its factory value. If an ACS modifies the value of this parameter, it SHOULD be prepared to accommodate the situation that the original value is restored as the result of a factory reset.
PeriodicInformEnable	W	Whether or not the CPE MUST periodically send CPE information to the ACS using the Inform method call.
PeriodicInformInterval	W	The duration in seconds of the interval for which the CPE MUST attempt to connect with the ACS and call the Inform method if PeriodicInformEnable is true.
PeriodicInformTime	W	An absolute time reference in UTC to determine when the CPE should initiate the Inform method calls. Each Inform call must occur at this reference time plus or minus an integer multiple of the {{param|PeriodicInformInterval}}. A zero dateTime value (0000-00-00T00:00:00) indicates that no particular time reference is specified. That is, the CPE may locally choose the time reference, required only to adhere to the specified
ParameterKey	R	ParameterKey provides the ACS a reliable and extensible means to track changes made by the ACS. The value of ParameterKey MUST be equal to the value of the ParameterKey argument from the most recent successful SetParameterValues, AddObject, or DeleteObject method call from the ACS.
ConnectionRequestURL	R	HTTP URL for an ACS to make a Connection Request notification to the CPE.
ConnectionRequestUsername	W	Username used to authenticate an ACS making a Connection Request to the CPE.
ConnectionRequestPassword	W	Password used to authenticate an ACS making a Connection Request to the CPE. When read, this parameter returns an empty string, regardless of the actual value.
Device.GatewayInfo.		**This object contains information associated with a connected Internet Gateway Device.**
ManufacturerOUI	R	Organizationally unique identifier of the associated Internet Gateway Device. An empty string indicates that there is no associated Internet Gateway Device that has been detected.
ProductClass	R	Identifier of the product class of the associated Internet Gateway Device. An empty string indicates either that there is no associated Internet Gateway Device that has been detected, or the Internet Gateway Device does not support the use of the product-class parameter.
SerialNumber	R	Serial number of the associated Internet Gateway Device. An empty string indicates that there is no associated Internet Gateway Device that has been detected.
Device.LAN.		**This object contains parameters relating to IPbased LAN connectivity of a device.**
AddressingType	R	The method used to assign an address to this interface. Enumeration of: “DHCP” “Static”

IPAddress	R	The current IP address assigned to this interface.
SubnetMask	R	The current subnet mask.
DefaultGateway	R	The IP address of the current default gateway for this interface.
DNSServers	R	Comma-separated list of IP address of the DNS servers for this interface.

Annex L New Event package for SIP SUBSCRIBE /NOTIFY (informative)

- Event: oipf-spdlist;DomainName="koreatelecom.co.kr".

The newly created event name SHALL be "oipf-spdlist"

  - Also, the *DomainName* parameter can have either "*ALL*" or a certain service provider's domain name. If the value of *DomainName* parameter SHALL be "*ALL*", IPTV SP Discovery FE should return all SP discovery information of all service providers. However, if the *DomainName* value SHALL be a specific domain address, the SP Discovery FE should return the request specific SP discovery information.

- Extending the existing Accept-Encoding:

  - The value of "Accept-Encoding" can be either "gzip" or "xml-oipf-bim". When OITF requests with "xml-oipf-bim", the SP Discovery FE should return the SP Discovery xml document which SHALL be encoded with BiM.

- New Event package for SIP NOTIFY request

  - The Event header of the SIP NOTIFY request SHALL be set with "oipf-spdlist".

  - Event: oipf-spdlist;

  - The Content-Encoding SHALL be either "gzip" or "xml-oipf-bim".

  - The "effective-by" parameter for the event header SHALL be set to 0.

  - The Content-Type SHALL be "application/vnd.oipf.spdlist+xml".

Annex M Schema Extension for FLUTE FDT

M.1 Namespace

```
<schema xmlns:tns="urn:oipf:protocol:fluteFDT:2009"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:fl="http://www.example.com/flute"
targetNamespace="urn:oipf:protocol:fluteFDT:2009" elementFormDefault="qualified"
attributeFormDefault="unqualified">
<!-- schema filename is  protocol-fluteFDT.xsd -->
```

M.2 Import Namespace and schema

```
<import namespace="http://www.example.com/flute"
   schemaLocation="imports/Flute_FDT.xsd"/>
```

M.3 Extension of FDT Attributes

When FLUTE is used for delivery of objects via multicast the FDT-Instance XML structure is extended using the extension mechanism defined in [RFC3926].

The following attributes are added to the FDT-Instance element, after the standard attributes.

```
<complexType name="OIPFCommonFDTAttributes">
  <complexContent>
    <extension base="fl:CommonFDTAttributes">
      <attribute name="Tags" type="string" use="required">
        <annotation>
          <documentation>
            This string value contains multiple tags, delimited with a
            semicolon (";"), that describe the File
            e.g. "Tag1;Tag2;Tag Three;"
          </documentation>
        </annotation>
      </attribute>
      <attribute name="Priority" type="positiveInteger" use="optional"
         default="10"/>
    </extension>
  </complexContent>
</complexType>
```