ETSI TR 126 939 V18.0.0 (2024-05)

Universal Mobile Telecommunications System (UMTS);
LTE; 5G;
Guidelines on the Framework for Live Uplink Streaming (FLUS)
(3GPP TR 26.939 version 18.0.0 Release 18)

Reference
RTR/TSGS-0426939vi00

Keywords
5G,LTE,UMTS

ETSI

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

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

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Important notice

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

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

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

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

If you find a security vulnerability in the present document, please report it through our
Coordinated Vulnerability Disclosure Program:
https://www.etsi.org/standards/coordinated-vulnerability-disclosure

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.
The content of the PDF version shall not be modified without the written authorization of ETSI.
The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2024.
All rights reserved.

Intellectual Property Rights

Essential patents

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

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

Trademarks

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

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Legal Notice

This Technical Report (TR) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities. These shall be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between 3GPP and ETSI identities can be found under https://webapp.etsi.org/key/queryform.asp.

Modal verbs terminology

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

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

Contents

Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

Y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

Introduction

The present document describes the ways to use the Framework for Live Uplink Streaming to setup services that allow the end user to stream live feeds into the network or to a second party.

1 Scope

The present document describes how to use the Framework for Live Uplink Streaming (FLUS) to stream live feeds to the network or to a second party. It describes the usage of both variants: the IMS-based and the non-IMS-based framework to carry regular 2D and 360 degrees video feeds. It also describes a set of instantiations for the non-IMS-based solution as the FLUS User Plane has been left for the discretion of implementations to support a diversity of requirements that require different instantiations of the user plane.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 26.238: "Uplink streaming".

[3] ISO 14496-12: "Information technology – Coding of audio-visual objects – Part 12: ISO base media file format".

[4] ISO 23000-19: "Information technology – Coding of audio-visual objects – Part 19: Common media application format (CMAF) for segmented media".

[5] 3GPP TS 23.401: "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access".

[6] 3GPP TS 23.501: "System Architecture for the 5G System (5GS)".

[7] ISO 14496-12: "Information technology – Coding of audio-visual objects – Part 12: ISO base media file format".

[8] ISO 23008-1: "Information technology – High efficiency coding and media delivery in heterogeneous environments – Part 1: MPEG media transport (MMT) ".

[9] ISO 23008-1: 2nd Edition AMD2, "Enhancements for Mobile Environments".

[10] IETF RFC 6455: "The WebSocket Protocol".

[11] IETF RFC 5234 (2008): "Augmented BNF for Syntax Specifications: ABNF", D. Crocker, P. Overell.

[12] IETF RFC 6817: "Low Extra Delay Background Transport (LEDBAT) ".

[13] ISO 23000-19: "Common Media Application Format for Segmented Media (CMAF) ".

[14] IETF RFC 7230: "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing".

[15] IETF RFC 7540: "Hypertext Transfer Protocol Version 2 (HTTP/2) ".

[16] 3GPP TS 26.114: "IP Multimedia Subsystem (IMS); Multimedia Telephony; Media handling and interaction".

[17] 3GPP TS 26.247: "Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)".

[18] 3GPP TS 26.244: "Transparent end-to-end packet-switched streaming service (PSS); 3GPP file format (3GP)".

[19] 3GPP TS 23.203: "Policy and Charging Control Architecture".

[20] DVB CM-AVC Report on Low-Latency Live Service with DASH.

[21] "Facebook (A)Live? Are live social broadcasts really broadcasts? ", by A. Raman, G. Tyson and N. Sastry, WWW 2018: The 2018 Web Conference, April 23-28, 2018, Lyon, France, https://arxiv.org/pdf/1803.02791.pdf.

[22] 3GPP TS 26.347: "Multimedia Broadcast/Multicast Service (MBMS); Application Programming Interface and URL".

[23] 3GPP TR 26.985: "Vehicle-to-everything (V2X) media handling and interaction".

[24] ISO/IEC 23090-8 Information technology — Coded representation of immersive media — Part 8: Network based media processing

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in 3GPP TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in 3GPP TR 21.905 [1].

FLUS session: A logical association between a source and a sink within which media content can be sent from the source to the sink.

Media session: A subset or part of a FLUS session including the duration to establish the media session, the time period during which media content can be sent from FLUS source to FLUS sink and the duration to terminate the media session.

Media stream: The content sent from a FLUS source to a FLUS sink within a media session.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in 3GPP TR 21.905 [1].

FLUS	Framework for Live Uplink Streaming
MCC	Mobile Country Code
MNC	Mobile Network Code
NBMP	Network Based Media Processing

4 FLUS Overview

FLUS defines a FLUS source entity and a FLUS sink entity that can support point-to-point transmission of speech/audio, video, and text. It defines media handling (e.g., signalling, transport, packet-loss handling, and adaptation). The goal is to ensure a reliable and interoperable service with a predictable media quality while allowing for flexibility in the service offerings.

A FLUS source entity, which may be embedded in a single UE, or distributed among a UE and separate audio-visual capture devices, may support all or a subset of the features specified in the present document.

When used as a generic framework, only the F-C procedures for establishing the FLUS session are required to be supported by the source and sink entities, and no other feature or procedure specified in the present document is mandated. Impact on the service quality and network capacity is left to the discretion of the implementation and the service utilizing the framework. For example, configuration of media formats and codecs follows the requirements of the respective service.

When offered as part of a 3GPP IMS/MTSI service, the source and sink are required to support the IMS control plane and media plane procedures, and the service quality is determined by the MTSI service policy.

When offered as part of a 3GPP non-IMS/MTSI instantiation, the service may make use of the Network Assistance feature for uplink services. For the IMS instantiation of FLUS, the source and sink may make use of Access Network Bitrate Recommendation (ANBR) functionality as defined in clause 10.7 of TS 26.114 [16].

The present document provides guidelines for the usage of FLUS and describes different user plane instantiations that can be used with FLUS.

5 Guidelines for IMS-based FLUS

Guidelines for the usage of FLUS in the IMS-based operation mode are not provided in the present document.

6 Guidelines for non-IMS-based FLUS

6.1 Use Case: Sharing to a Social Network Service

A user shares a 360 degree video that is being captured through a VR camera and sent as a fish eye, side-by-side 2D video. The 360 video stream is shared with a FLUS Sink in the network that relays the stream to a popular social network service (SNS).

6.2 Use Case: Live uplink video stream from drones or moving vehicles

The media producer for an event is using drone-mounted-360 cameras or other moving vehicles like F1 cars, sailing boats or bicycles to capture scenes from more innovative angles. The drone is flown using line of sight, i.e. the drone pilot has direct visual contact to the drone. Other vehicles may have the driver / pilot on-board.

The live video is streaming to the live ingest server and then used together with other camera feeds in a live TV broadcast.

In particular for battery powered cameras, it may be beneficial to avoid processing like 360 video stitching on the device. Instead, it may be beneficial to leverage network based post processing functions, e.g. multiple video streams are transmitted and the stitching function is executed in the network.

Use-case example: An event-organizer plans to use multiple drone mounted-cameras to capture live video from an event. All live video streams should be routed to an editing facility, where a program director decides on the sequencing of live video into a single linear program. The media source of each drone is configured with their own target quality (bitrate) and target delay. Each media source is configured with a unique media sink so that the program director can identify each media source.

This use-case is generic in the sense that the camera is not limited to be "drone mounted" but can be mounted to any device, vehicle or stationary object.

6.3 Use Case: Breaking-News reporter

A news corporation uses 5G and mobile equipment to speed up and simplify their breaking-news operations. Either professional cameras are equipped with 5G uplink streaming modems, or regular smartphones (with external

microphones) are used for video capture. The universally available 3GPP coverage is used to stream the live video (with configurable, low delay) from the breaking news scene into the broadcast operation studio.

A news corporation negotiates a Service Level Agreement (SLA) with an operator so that a set of reporters can do either sequential or simultaneous live reports. The general frame agreement between the news corporation and the MNO foresees that each reporter can determine its own maximum video quality (measured in bits per second). Each reporter should set their own quality, but some reporters are allowed to provider higher quality (i.e. use higher bitrates) than others.

This use case illustrates well the benefits and usage of Network Assistance for FLUS, whereby a mobile professional news team that is sent out to cover a sudden news situation. The team consists of the camera/sound man and the reporter. They arrive at the scene on their motorbike. The camera/soundman unpacks the camera, powers it on, and mounts it on his shoulder ready to shoot as soon as the reporter has made a quick assessment of the scene and is ready to report back to the TV studio.

The UE logs in to the network and establishes the secure, reliable, high bandwidth 5G connection with the TV studio, under the terms of the special subscription, or Service Level Agreement (SLA), that the TV station has with the incumbent 5G MNO in the region.

While the issue of SLA is out of scope of the FLUS specification, it is useful to consider two kinds of SLA that could be relevant here:

1) SLA with guarantees for certain uplink bitrates – for this the MNO provisions RAN and network capacity in order for the TV station to be able to use FLUS to upstream their preferred media stream format reliably, anywhere where the MNO's network is reachable.

2) SLA with higher prioritisation of uplink media streams – for this the MNO agrees to prioritise uplink media streams bound for the TV station, over those of other users that do not have the SLA in place. This facility could also be included in the first type of SLA.

In the present use case, it is assumed the connection is ad-hoc in nature, so there is no guaranteed QoS from the MNO in this case. This corresponds to the second kind of SLA mentioned above. The objective of the news team is to deliver the best possible quality version of coverage that is possible in that location with the network capabilities and conditions at hand, allowing for possible variations during the coverage. In other variants of this use case, there might be provision for a particular quality level of audio/video stream to be delivered, for example using a QoS framework.

After the camera unit has powered up, it proceeds through the sequence of interactions with the network, so that it is ready to start streaming the best possible quality version of audio/video coverage under the current local network conditions when live coverage starts.

6.4 Use Case: Immersive media conversations

In this scenario, streams of 360 video and multi-channel audio are transmitted from a media sender to a media receiver, which at the receiver side, are projected on a screen or a HMD, and played out with loudspeakers or a headphone. In the other direction, video bit-streams of lower quality or resolution are transmitted to show the sender how the far-end user is watching and hearing the video and audio. A session is used to provide two-way real-time voice conversation. The 360 video and multi-channel audio are synchronized but arrives slightly later than the speech frames captured at similar times.

6.5 Use Case: Feedback-enabled Control of Uplink Streaming

6.5.1 Use Case Description

This use case is based on the scenario described in clause 6.1, regarding a user sharing a 360-degree video clip captured by a VR camera with viewers, via the FLUS service provider which processes and subsequently forwards the stream to a social network service (SNS).

The SNS provider is aware of the size of the audience that is receiving the uplinked 360-degree video stream. The SNS provider needs to be able to control the uplink streaming session according to the size if the audience. For example, the FLUS source is controlled to not send content on the uplink, or only a small initial portion, until there is at least one viewer of his VR streaming content. Such control of uplink transmission may be performed in accordance to user

preference settings maintained by the SNS provider. In addition, the MNO, either in the role of the SNS provider or the FLUS service provider, has awareness of network conditions such as core network congestion, access network (3GPP RAN and/or other access network technologies) availability and status, UE location relative to the serving base station, and downlink distribution technology employed for delivering the uploaded media to viewers. The MNO needs to be able to control the uplink streaming session in accordance with these types of information.

An additional aspect to this use case is where the MNO is involved in managing network resources in connection with FLUS-based services. The MNO needs to be able to dynamically adapt to conditions in both the core network and the RAN to support optimal overall network performance, for example in throughout, delay and fairness for all users wishing to send data on the uplink. Some dynamic properties of the UE might come into play, for example the relative location of the UE in the serving cell.

The business entity corresponding to the SNS provider could be either a 3rd-party application service provider or an MNO.

The business entity corresponding to the FLUS service provider is an MNO.

6.5.2 Working Assumptions

The following working assumptions are applicable to the use case described in clause 6.5.1:

- The end user of the FLUS source wishes to avoid or minimize unnecessary uplink streaming traffic, for example, defer live uplink streaming or send only a small initial portion of the content, in the absence of viewership.
- The SNS provider has information on the actual or predicted viewership for any given instance of uplink streaming content.
- The FLUS services provider would like the capability to delay or defer streaming content upload by new FLUS sources when it encounters congestion or a high work load in core network processing of incoming media from existing FLUS sources.
- The FLUS service provider has knowledge of access network information and based on that information would like the capability to influence the data rate and/or QoS level of uplink transmission for an existing and active FLUS session.
- The FLUS service provider has knowledge of the location of the FLUS source relative to its serving base station and based on that information would like the capability to influence the data rate and/or QoS level of uplink transmission for an existing and active FLUS session.

6.5.3 Recommended Requirements

The following recommended requirements are derived from the use case description and working assumptions in clauses 6.5.1 and 6.5.2, respectively:

- It ought to be possible for the SNS provider and/or FLUS service provider to influence the uplink streaming behavior of the FLUS source including the properties of its streaming media content (such as video quality) due to, for example, the presence of congestion or high processing load in network-based media processing, access network related conditions, or location of the FLUS source relative to its serving base station.
- It ought to be possible for the SNS provider to utilize information on the measured or expected audience of, or interactive engagement of viewers with, the uplink streaming content in order to adapt the uplink streaming behavior of the FLUS source including the properties of its media content, in accordance with SNS provider policy. These might take into account user preferences within the scope of the corresponding SNS.

6.5.4 Gap Analysis

The following gap is identified with regards to meeting the recommended requirements as described in clause 6.5.4:

- The existing FLUS system architecture does not describe or depict the sources and types of information (application level, core network and/or access network related status/conditions), nor the suitable propagation of

that information. The sources, types and means of propagation of such information are deemed relevant to the SNS provider and/or MNO for controlling or guiding uplink transmission behavior of the FLUS source.

6.6 Use Case: Media production

6.6.1 General

The general use case for media coverage of an event, for example a sports match or a musical performance, is for cases where media capture equipment is installed specifically for the coverage of the event. Media capture equipment will be connected via cabling by default, commonly coaxial cabling, optical fiber, or Gigabit-Ethernet. Where it is more convenient and feasible, some of the capture equipment can be connected wirelessly, using FLUS as foreseen in the present use case. Wireless connections offer the possibility to give spectacular vantage positions for the coverage, from places not easily accessible via cabling, or for more flexible and rapid mobile coverage, for example from a shoulder-mounted camera, from a participant wearing a body-mounted camera, or from mobile unmanned vehicles like drones. The minimalistic media production use case consists of a single remote camera feed that uplinks content via FLUS, which corresponds to the breaking news reporter use case above.

The media production use case calls for the highest possible quality of audio and video content that is delivered via FLUS, but not at the sacrifice of reliability of delivery. For example, video content is preferably only lightly compressed, including only intra-frame coding. This allows the most flexibility and preservation of quality with subsequent production operations, for example mixing and editing. However, for practical purposes, the media production use case could have several production levels available – highly compressed, lightly compressed, uncompressed. Several media formats could be defined within each of these production levels.

In some production situations, it is necessary for the production centre to communicate in real time with camera personnel, to execute direction for the coverage. Communication can be individually to each FLUS Source, and/or among all participants, both at the production centre and among the capture equipment personnel at the same time. It could be beneficial to include this in the FLUS system since the same physical medium is used for communications as for uplink stream delivery.

Typically there are several FLUS sources with their media capture devices that deliver media streams or files to be uplinked using FLUS. As each FLUS source is set up and prepared for the event, it proceeds, either by manual trigger or autonomously, to register with the FLUS sink and remote production centre. Once a FLUS source is known to the FLUS sink, and if applicable, also to the remote production centre, full control of that device and the FLUS sessions involving that FLUS source should be given to the FLUS sink and production centre behind it, if applicable.

The media production use case is distinguished from the breaking news reporter use case in that the former preferably can rely on adequate network resources to deliver the foreseen number and quality of audio-visual content essences during the production, and does not use an operator’s open network that is available at the location, without any special facilities for ensuring adequate network resource for the media feed. Hence it is expected that it will make use of private network resources that are provided specifically for media production purposes. The breaking news reporter use case may also be able to rely on such facilities but in many cases is expected to work also on any locally available operator network.

There are many different kinds of media production scenario that can make use of FLUS to deliver content originating from the coverage of a situation or event. Some typical examples of media production use cases are described in sub-clause 6.6.2 below.

Consideration of such a variety of variants of the use case results in recognized needs for additional functionality in FLUS. This is described under "gap analysis" in the sub-clause 6.4.3, and summarized in sub-clause 6.7.4 under "recommended requirements".

6.6.2 Variants of the use case

6.6.2.1 Remote production

Remote production means the case where one or more media capture source devices deliver media via FLUS to a production facility, for example a news organization facility or TV station, via one or more FLUS sink instantiations in the network. This variant includes the "breaking news reporter" use case that typically deploys a single capture source.

The delivered media assets can consist of any formats defined by 3GPP, hence for video they can be 2D or 360-degree video streams. Especially for live production, the latency from FLUS source(s) to the FLUS sink, and on to the production facility, should be as low as possible, and not exceed a certain threshold.

When multiple source capture devices are deployed, typically they need to be synchronized to aid production in a live/linear production flow.

6.6.2.2 Local pre-production

This variant revolves around one or more contribution streams from FLUS sources being delivered to the FLUS sink that is located at or in proximity of the event being covered. Some of the production process is performed locally, for example ensuring consistency between all video and audio sources, or directing camera shots. One or more media stream feeds are provided via a further FLUS source to the FLUS sink in the network, for final production remotely from the event. A cable or fibre connection can also be used to carry the output media stream feeds.

FLUS can be used locally to capture media streams from some or all of the equipment in use, but also a separate FLUS connection could be used to deliver the (pre-)produced media stream(s) to the network, either to a final production centre or for re-distribution.

In some production situations, a media feed is sent back to the FLUS Source device for monitoring purposes. For example, a cameraman might have a small monitor attached to the side of the camera to view the output stream of the ongoing production.

6.6.2.3 Local complete production

In addition to the previous variant, this variant involves final production on-site at the event, meaning that a single or multiple media feeds are delivered to the network, either via a FLUS connection or via a cable/fibre feed, to the network for onward distribution.

6.6.3 Gap analysis

With professional media production systems, sometimes the production personnel, e.g. the camera operators, receive one or more feedback media streams, for monitoring purposes. One example is when the final production output is relayed back to the production personnel at each capture device. While the feedback stream is not an uplink per se, it could be beneficial to include it in the FLUS system, so that it can be managed logically within the same framework as the uplink media stream(s) delivered by the same FLUS Source. In principle there could be multiple return media streams.

With professional media production systems it is common practice that production personnel, for example camera operators, communicate with the production centre, and/or among themselves, during the production. Communications need to have low latency, so that production personnel can react swiftly to the director’s instructions. In addition to verbal communications, a "tally" facility would also be of benefit within the FLUS system, whereby the production centre is able to signal to a capture device that its output is currently live (on air), or about to go live – so-called "preview" state – relayed to production personnel and talent (e.g. reporter), e.g. via a red light on the camera. Such communication media streams could be realised outside the FLUS system, but it may be advantageous to include these streams within the FLUS system.

In many media production scenarios it could be beneficial for the production centre to have full control over FLUS sessions, as well as of the capture devices themselves and the media stream(s) that are delivered. Currently in the FLUS specification it is possible only for the FLUS source to control FLUS sessions.

The media production use case variants illustrate the various kinds of media capture devices that can come into play. In many situations it could be beneficial to be able to characterise the devices using common terminology, so that the production centre can more easily organize and work with capture devices from various providers, by first recognizing their capabilities and status.

The following set of characteristics attributed to FLUS source devices are envisaged to be useful:

- Available video format(s)
- Available audio format(s)
- Available ancillary streams: subtitles / captions, content metadata

- Connectivity: RAN, wired – and which system/s
- Control capability: manual, remote, none – and which system/s
- Mobility - one of: fixed, on foot, ground vehicle, airborne vehicle, surface water vehicle, underwater
- Power – one of: Battery, battery with autonomous charging, mains

Some of these properties can be dynamic and change during an event, so it will be beneficial to be able to enquire the status or receive updates from devices that change their status during the event.

6.6.4 Recommended requirements

Based on the gap analysis, the following features are seen to be directly relevant to the FLUS system, so should be considered for inclusion in TS 26.238 [2]:

- Control of FLUS sessions by the FLUS sink;
- FLUS source capability discovery;
- Characterization of FLUS sources.

The following features, while being required for the media production use case, are arguably not within the scope of the FLUS specification as it is defined currently. It should be considered how to accommodate these features, if not within the scope of FLUS directly, then either as a defined service, or otherwise:

- Return media stream(s) for the FLUS source;
- Communication facility to and among media production personnel, for example, verbal communications and tally.

6.7 Use Case: Fisheye omnidirectional video sharing

One type of camera being widely used today for capturing of 360 video is fisheye cameras consisting of one or more fisheye lens. In this scenario, streams of 360 video with one or more fisheye cameras are transmitted from a media sender to a media receiver. The fisheye lens are wide-angle camera lens that usually captures an approximately hemispherical field of view and produces a circular image. Circular images captured by multiple fisheye lenses of a 360 camera typically cover all directions around the centre point of the camera set or camera device. An example of a fisheye video containing two circular images captured by two fisheye lens is shown in the below Figure 1.

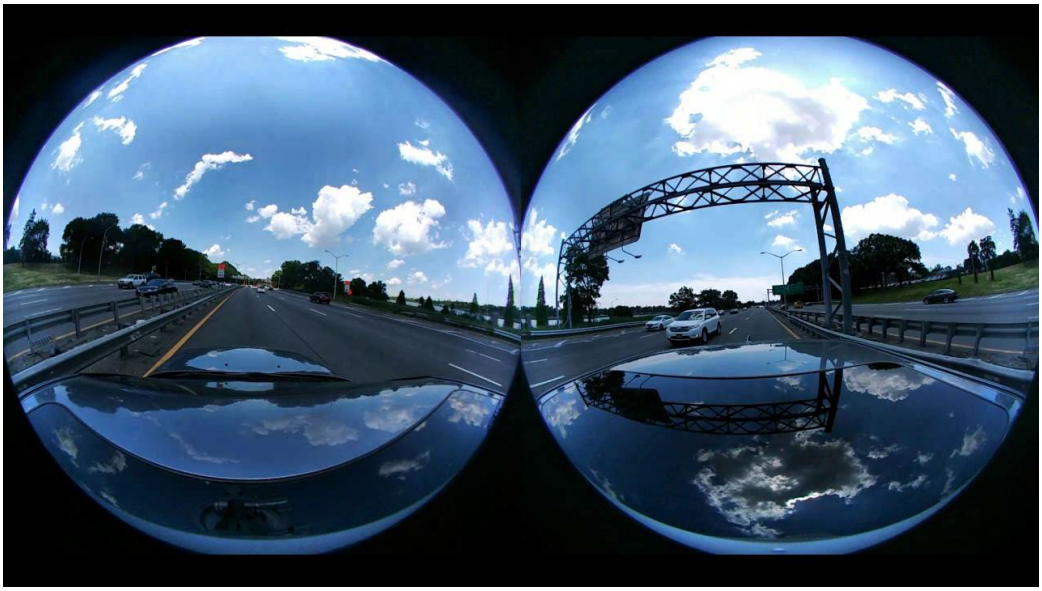


Figure 1: Two fisheye images per video picture at a media sender

When the fisheye video is captured at a media sender, there is usually at most very little overlapping areas of the video captured by the lens. Therefore, the projection does not provide much benefit to remove or reduce redundant overlapping area even though it requires additional processing for the projection. When the fisheye video is directly encoded and transmitted, the fisheye video is decoded and the part corresponding to the user viewport is provided, with or without the projection.

For sharing to social network service, a user is sharing a 360 degree video that is being captured through a VR camera and sent as a fish eye. The video stream is shared with a FLUS Sink in the network that relays the stream to a popular social network service (SNS).

Once the user selects to start sharing, the UE discovers an appropriate FLUS Sink that supports the specific SNS and that can stitch the fish eye into a 360 video and transcode the content to match the distribution format.

Note: the use case is not limited to just capturing video but can also capture spatial audio. When both are captured at the FLUS source, both video and audio media are streamed to the FLUS sink.

6.8 Use Case: Live uplink streaming for V2X Services

6.8.1 Use Case Description

In this scenario, streams of 2D or 360 video are transmitted from FLUS source, e.g., a local camera mounted on the roadside, towards a remote server as FLUS sink. The video, and possibly audio, taken of the VRU (vulnerable road user) along a road need to be discovered by the object-detection server and then the object information needs to be communicated to vehicles with different LoAs (Level of Automations). A session is established between the FLUS source and sink in which the end-to-end latency between the camera on the FLUS source, the sink on the server, and communicating the object information to the vehicle should be lower than a certain threshold so that the VRUs are monitored in a real-time manner [23].

6.9 Use Case: Network Based Media Processing

In this scenario, the network-based media processing based on the NBMP standard [24] is used for media processing of the content in the network. The use-case consists of several use cases.

1 Use Case Description

2 Live streaming

Kim is subscribed to an Application for live streaming of captured videos from her everyday life. Based on the previous number and diversity of Kim's usual audience (e.g. close friends), the Application has an "audience codecs-rates" profile which represents the typical number of streams needed based on Kim's previous streaming sessions. Kim starts the session live stream session. While Kim is uploading a single stream using FLUS, the server Application is commanding the running of multiple transcoders based on Kim's audience codec-rates profile. If new users join Kim's streaming session which could not be supported with the current codecs-rates, the Application may add more transcoders to add to multirate streaming in the session.

3 Variations

1. Capabilities:
    a. There are sufficient resources available at FLUS Sinks, so there is no need to check whether the picked FLUS Sink has the required real-time multi-rate transcoding capabilities.
    b. Available FLUS Sinks might have limited capabilities. The server application finds a Sink capable of running the transcoding session.
2. Server or Device Application
    a. The UE's Application is responsible for setting up the FLUS and NBMP session, as well as managing the audience codecs-rates profile.
    b. The UE's Application is responsible for setting up the FLUS session. The Sever Application is responsible for setting up the NBMP session.

4 Preconditions

1. On the device:

a. A 3GPP supported encoder is installed.

b. UE's Application is installed which supports NBMP Client functionalities.

c. A 3GPP FLUS Source is installed.

2. On the network,

a. One or more FLUS Sinks are installed that one or more of them supports NBMP Workflow Manager functionality.

b. One or more FLUS Sinks are installed, that run various instances of decoding/encoding or transcoding, and the decoder/encoder/transcoder are described as NBMP Functions in an NBMP repository.

c. A FLUS Sink may have limited capabilities, i.e. the codecs it supports and/or the number of concurrent transcoders.

d. The FLUS Sink's load may vary dynamically due to the other parallel network processing sessions.

e. Network storage is available to store the encoded content for time-shifted streaming.

5 Requirements in terms of Capabilities and QoS/QoE Considerations

1. Capabilities
   - Discovering of FLUS Sink's network media processing capabilities if needed
   - Setting up the FLUS and NBMP sessions
   - Start the Sessions
   - Change the workflow by adding more transcoders if needed during the session
   - Real-time transcoding and packaging to different codecs and different bitrates
   - Offloading an originally selected FLUS Sink to a new FLUS Sink to handle additional needed processing that cannot be accommodated by the original FLUS Sink
2. KPI
   - Supporting the use-case
   - Minimum extension of FLUS and NBMP Standard, preferably none.

6 Rich on-demand video streaming

Khloe, Kim's sister, is subscribed to an Application for uploading her videos, enhancing them, and adding rich features to them before publishing them as on-demand content. Example of enhancements and added rich features are:

1. Enhancing the quality of the video with e.g. noise removal, white balance correction, color correction, prebuilt filters, etc.
2. Indexing the content
3. Creating thumbnail navigation
4. Extracting subtitle from audio and add it to the video
5. Translation and adding multiple language subtitles

6. Adding interactive tags to the shops, restaurants, and other service stores in the scene

7. Detecting any unintended improper shots and marking them for Khloe to review

Based on the previous number and diversity of Khloe's usual audience (e.g. close friends), the Application has an "audience codecs-rates" profile which represents the typical number of streams needed for streaming sessions. Depending on the service, the encoding can be done ahead of time, or on-fly when is requested by an audience. The on-demand content becomes available with a maximum delay from the end of upload. This maximum delay is defined by the Application in Khloe's profile.

7 Variations

1. Capabilities:

   a. There are sufficient resources available at FLUS Sinks, so there is no need to check whether the picked FLUS Sink has the required real-time multi-rate transcoding capabilities.

   b. Available FLUS Sinks might have limited capabilities. The server application finds a FLUS Sink capable of running the transcoding session.

   c. Required FLUS Sink resources vary based on the number and sophistication of rich features and the processing time defined by the user's profile.

2. Server or Device Application

   a. The UE's Application is responsible for setting up the FLUS and NBMP session, as well as managing the audience codecs-rates profile.

   b. The UE's Application is responsible for setting up the FLUS session. The Server Application is responsible for setting up the NBMP session.

8 Preconditions

1. On the device:
   a. A 3GPP supported encoder is installed.
   b. UE's Application is installed which supports NBMP Client functionalities.
   c. A 3GPP FLUS Source is installed.
2. On the network,
   a. One or more FLUS Sinks are installed that one or more of them supports NBMP Workflow Manager functionality.
   b. One or more FLUS Sinks are installed, that run various instances of decoding/encoding or transcoding, and the decoder/encoder/transcoder are described as NBMP Functions in an NBMP repository.
   c. A FLUS Sink may have limited capabilities, i.e. the codecs it supports and/or a limited number of concurrent encoding/transcoding sessions.
   d. A FLUS Sink's load may vary dynamically due to the other parallel network processing sessions.
   e. Network storage is available to store the encoded content for time-shifted streaming.

9 Requirements in terms of Capabilities and QoS/QoE Considerations

1. Capabilities
   - Discovering of FLUS Sink's network processing capabilities if needed
   - Setting up the FLUS and NBMP sessions
   - Start the Sessions
   - Change the workflow by adding more transcoders if needed during the session
   - Real-time transcoding and packaging to different codecs and different bitrates
   - Offloading an originally selected FLUS Sink to a new FLUS Sink to handle additional needed processing that cannot be accommodated by the original FLUS Sink
2. KPI

- o Supporting the use-case
- o Minimum extension of FLUS and NBMP Standard, preferably none.

10 Assumptions and requirements

11 NBMP in the current FLUS architecture

This clause describes the use of NBMP [24] in the current FLUS architecture. An overview of NBMP architecture can be found in [2] Annex B.

12 Mapping between system components

Based on the functional definitions of different NBMP system components, the following mapping between NBMP components and FLUS system components can be made, as shown in Table 6.9.2.2-1.

Table 6.9.2.2-1. Mapping from NBMP components to FLUS components

System Component	Description
NBMP Client	• FLUS Control Source inside FLUS Source: The FLUS Source uses the F-C interface to set up workflows at the FLUS Sink as described in TS 26.238 and TR 26.939. F-C uses Workflow API as defined in ISO/IEC 23090-8 for this procedure • Non-colocated Control Source: The Non-colocated control source outside the FLUS source, described in clause A.1.3 of 3GPP TS 26.238, can take the role of NBMP Client and use the F-C interface to configure workflow at the FLUS Sink. • FLUS Control Source inside the Remote Control Device: The Control Source inside the control device as described in TS 26.238 clause A.2.2 can take the role of NBMP Client and use the F-C interface to configure workflow at the FLUS Sink. • External Application Server: An Application Server in trusted DN or external DN can take the role of NBMP Client. • Application on UE (UA): An Application on UE (UA) can take the role of NBMP Client.
NBMP Media Source	• Media source inside FLUS Source: The media source inside FLUS Source assumes the role of NBMP Media Source.
NBMP Workflow Manager	• FLUS Control Sink inside FLUS Sink: The control sink inside FLUS Sink can take the role of NBMP workflow manager. The FLUS Control Sink sets up post-processing and distribution functions as described in TR 26.939 clause A.1 and A.2 in one or more NBMP media processing entities. • External Application Server: An Application Server in trusted DN or external DN can take the role of NBMP Workflow Manager and receive a workflow description from an NBMP Client.
NBMP Task	• FLUS Media Sink inside FLUS Sink: The FLUS Media Sink function of FLUS Sink assumes the role of NBMP Task to ingest content from FLUS Source using the F-U interface. The ingested content can then be sent to post-processing and distribution functions in other NBMP Tasks in the workflow setup by the workflow manager.

NBMP Function Repository	None

13 API considerations

Table 6.9.2.3-1 shows the mapping between different NBMP API and FLUS API.

Table 6.9.2.3-1. Mapping NBMP API to FLUS API

API	Description
Workflow API	• Uplink Streaming Control Interface as defined in TS 26.238 clause 7 is to be used for NBMP Workflow API
Task API	• Currently outside the scope of FLUS specification (TS 26.238)
Function Discovery API	• Currently outside the scope of TS 26.238

14 Procedures

In the case in which NBMP sessions are managed through the FLUS control plane, the following procedures defined in TS 26.238 can be used or updated.

- Workflow Manager Discovery: Clause 7.2 of TS 26.238 describes the discovery procedure of FLUS Sink. This procedure can be used to discover a FLUS Sink that can act as an NBMP workflow manager as described in clause 4 of this contribution.
- Workflow Manager Capability Retrieval: Clause 7.3 of TS 26.238 describes capability retrieval of a FLUS Sink. This procedure can be used to retrieve workflow management capabilities at the FLUS Sink as described in clause 4 of this contribution.
- Workflow Establishment: Clause 7.5 of TS 26.238 is used for setting FLUS sessions between FLUS Source and FLUS Sink. This procedure can be used for setting up a workflow session at the FLUS Sink.
- Workflow Termination: Clause 7.6 of TS 26.238 is used for terminating FLUS sessions between FLUS Source and FLUS Sink. This procedure can be used for terminating a workflow session at the FLUS Sink.
- Workflow Modification: Clause 7.4.2 of TS 26.238 is used for modification of FLUS sessions between FLUS source and FLUS sink. This procedure can be used for modifying a workflow at the FLUS sink.
- Workflow Retrieval: Clauses 7.4.1 of TS 26.238 is used for retrieval of a FLUS session between FLUS Source and FLUS Sink. This procedure can be used for retrieving a workflow at the FLUS Sink.

7 FLUS User Plane Instantiations

7.1 Non-IMS-based User Plane Instantiations

7.1.1 Introduction

This clause describes a set of instantiations for the generic FLUS User Plane that is not based on IMS.

7.1.2 fMP4-based Instantiations

7.1.2.1 Introduction

All instantiations of this clause are based on the fragmented ISOBMFF [3] format which is profiled by CMAF [4]. The following description summarizes the used media format used in the present document:

1) Each media component is formatted as a CMAF Track.

2) Each CMAF Track starts with a CMAF Header followed by one or more CMAF Fragments. A CMAF Fragment contains one or more CMAF Chunks. Note that CMAF requires that only the first CMAF chunk of a CMAF fragment is constrained to be an adaptive switching point. All subsequent CMAF Chunks do not need to contain any service access point.

3) When CMAF Fragments contain more than one CMAF chunk, it is beneficial that the first CMAF Chunk of the CMAF Fragment is preceded by a SegmentTypeBox that includes the compatible_brands ‘cmfl’, ‘cmff’.

Note that the present document only considers the CMAF file format specific features.

7.1.3 fMP4 over MMTP Instantiation

7.1.3.1 General

MMTP is a transport protocol that supports the streaming of fragment ISOBMFF-formatted content using a dedicated payload format, the MPU payload format. Media data is streamed as a CMAF Header, followed by CMAF chunks for each media component separately. The CMAF Header is conformant to the MPU Header format and the CMAF Chunk is conformant to the MPU Fragment as specified in [7].

The FLUS Source and FLUS Sink use the MMTP protocol [7] over UDP, over DTLS/UDP or over WebSocket [10].

This instantiation is identified in the SDP by the protocol identifier: “MMTP/UDP”, “MMTP/DTLS/UDP”, or using a WebSocket URL (“ws” or “wss”) respectively, as specified in [9].

This instantiation is also identified in the F-C configuration by the following urn: “org:3gpp:flus:2018:instantiations:mmtp” or “org:3gpp:flus:2018:instantiations:mmtp-ws”, depending on whether using mmtp over UDP or over WebSocket.

Exactly one MMTP flow is used and each media component is sent using the MPU-mode, where each MPU conforms to the restrictions in clause 7.1.2.1.

The FLUS Source is required to maintain NTP synchronization, with a tolerance of ±20 ms, when setting the MPU_timestamp_descriptor and the MMTP delivery timestamp.

An fMP4 over MMTP/UDP or MMTP/DTLS/UDP session is described through an SDP file according to the constraints in [9]. The SDP is sent from the FLUS Sink to the FLUS source and includes exactly one media line that describes an MMTP flow in ***receive-only*** mode and with the target UDP port on which the MMTP flow is to be received at the FLUS Sink.

If the FLUS Sink is behind a NAT or Firewall, it has to ensure that the port is open (with correct NAT translation in place). It may also use known NAT traversal techniques, such as hole punching to establish the port binding at the NAT.

The source uses the MP table to describe the different components of the MMTP flow. The packet_id identifier is used to reference and map each component to the FLUS system.

7.1.3.2 MMTP Signaling

The content carried by an MMTP flow is described using the MP table, which is carried in an MPT message. The MP table with the relevant fields is provided in Table 7.1.3.2-1:

Table 7.1.3.2-1

Syntax	No. of bits
MP_table() {	
0xF	**8**
0x0	**8**
length	**16**
10111111b	**6**
number_of_assets	**8**
for (i=0; i<number_of_assets; i++) {	
Identifier {	
0x01	8
URL()	
}	
asset_type (4CC code)	**32**
0x0	**6**
1b	**1**
asset_location {	
0x01	**8**
Location {	
0x00	8
packet_id	16
}	
}	
asset_descriptors {	
MPU_timestamp_descriptor()	
}	
}	
}	

The MP table is a compact description of the MMTP flow. It is used to identify all components of the content and the MMTP sub-flows that carry them. It also carries asset descriptors, which are used to describe the different assets to which the actual components belong to.

7.1.3.3 Synchronization

FLUS supports different types of sources. In one scenario, the FLUS Source captures the content and streams it to the FLUS Sink. In another scenario, the content is coming from multiple devices but multiplexed at the FLUS Source and streamed to the FLUS Sink. Yet in another scenario, the content is coming from multiple FLUS Sources and is synchronized and multiplexed at the FLUS Sink.

In the former 2 scenarios, the FLUS Sink can assume that synchronization of the different components of the media stream(s) is taken care of by the FLUS Source. This can be achieved by setting the PTS appropriately in the fMP4 for each component and by using a common time baseline where all fMP4 tracks from all components have the same 0 origin.

In case the content is multiplexed at the source but actually created by other devices, a clock drift may result from different clocks at the generating devices and the clock at the FLUS Sink. The MMTP timestamp may be used by the FLUS Sink to estimate such drift. The MMTP packet timestamp is a the NTP short format and reflects the time of transmission of the MMTP packet and reflects the time when the MMTP stack queues the packet for transmission. It is generally used to measure estimate delay and delay jitter during the transmission of the MMTP packets.

If content is created by different FLUS Sources, synchronization needs to be performed at the FLUS Sink. The FLUS Sink uses the MMTP MPU_timestamp_descriptor() from each of the components, which aligns the start of a CMAF track to UTC time, which enables the FLUS Sink to align media from different FLUS Sources. The MPU_timestamp_descriptor provides a NTP 64-bit timestamp that describes the presentation time of the first media sample in presentation order in the CMAF Track.

The MPU_timestamp_descriptor is carried as part of the Asset descriptors in the MP table and it has the format as shown in Table 7.1.3.3-1:

Table 7.1.3.3-1

Syntax	Value	No. of bits	Mnemonic
MPU_timestamp_descriptor () {			
descriptor_tag		**16**	**uimsbf**
descriptor_length	**N*12**	**8**	**uimsbf**
for (i=0; i<N; i++) {			
mpu_sequence_number		**32**	**uimsbf**
mpu_presentation_time		**64**	**uimsbf**
}			
}			

In addition to the mpu_presentation_time, a field that carries the mpu_sequence_number, which provides an identifier of the current CMAF Track. This is important as the different sources may be forced to initiate a new CMAF Track during the transmission, e.g. after recovering from a failure.

7.1.3.4 Session Initiation and Description

Media Session initiation is triggered through the F-C procedure, where the session creation results in defining and returning the Entrypoint URL to the FLUS Sink. If MMTP/WebSockets is used then the Entrypoint URL points to a WebSocket URL and the MMTP sub-protocol is expected to be used. If MMTP/UDP is used, the Entrypoint URL points to an SDP file that is created by the FLUS Sink and used by the FLUS Source to start streaming data to the FLUS Sink. The SDP is generated by the FLUS Sink and includes exactly one FLUS media session, with recvonly attribute that indicates that media data is flowing uplink only.

An example of such SDP may look as follows:

```
v=0
o=user 6431641313 1 IN IP4 10.10.52.13
s=FLUS Session
t=1411639200 1427277600
a=source-filter: incl IN IP4 * 10.10.52.13
m=application 12345 MMTP/UDP 100 101 102
a=of:100 flowid=0 Signaling/PA
a=of:101 flowid=7623 MPU
a=of:102 flowid=7624 MPU
a=recvonly
```

7.1.4 fMP4-based Instantiation with HTTP Delivery.

7.1.4.1 General Description

This clause describes a FLUS Media Plane instantiation using a continuous sequence of CMAF Chunks [13] with HTTP Delivery, e.g. HTTP 1.1 [14] Chunked Delivery or HTTP 2.0 [15] Delivery.

This instantiation is identified in the F-C configuration by the following urn: “org:3gpp:flus:2018:instantiations:fmp4”. An additional F-C configuration option determines, whether HTTP 1.1 with Chunked Transfer Encoding, HTTP2.0 with TCP or HTTP 2.0 with other transport protocols such as QUIC should be used for the continuous upload of CMAF Chunks. Note, the HTTP version and the transport protocol may also be negotiated at connection setup. When using QUIC as transport protocol, the FLUS Source can fall back to TCP, when the QUIC session setup fails.

The CMAF Track used for this instantiation starts with the CMAF Header, followed by a sequence of CMAF Chunks.

The CMAF Header contains information around the number of tracks, the used codec, codec configuration and optionally static metadata for the upstreamed movie file.

When the FLUS session contains multiple media components, each component is formatted as a CMAF Track and upstreamed separately. A common presentation timeline is used across the different media components.

It is assumed that the FLUS Source provides only a single quality per media components to the FLUS Sink. It is assumed that this single quality is the highest quality and the FLUS Sink could create additional quality representations using a transcoder. However, it is in principle also possible that the FLUS Source provides multiple quality representations per media components.

The CMAF Chunks of the CMAF Track are continuously appended to a larger resource. The FLUS source is generally not adding ‘styp’ boxes, except if the immediately succeeding CMAF Chunk contains a service access point.

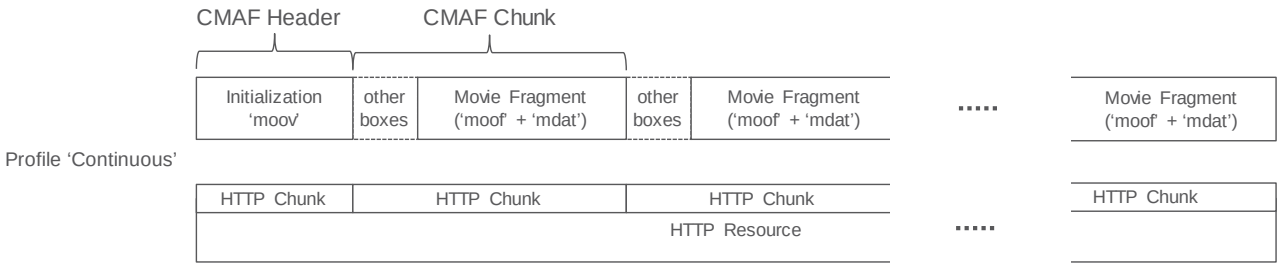


Figure 7.1.4.1-1: Illustration of Continuous Chunk Profiles (with mapping to HTTP 1.1 resources)

When a single FLUS Source streams multiple CMAF Tracks, a common media time line (i.e. decoding and composition timestamps) should be used across all CMAF Tracks. The FLUS source may insert wall clock timestamps using the Producer Reference Time Box (‘pfrt’box) into the CMAF Chunk stream. This may be beneficial, when streams from multiple FLUS Sources should be jointly post processed. The FLUS Source should be properly time-synchronized with the network, e.g. using EPS time synchronization derived from SIB16 (See TS 36.331) or NTP or other appropriate mechanisms.

This FLUS media instantiation focuses on the usage of the HTTP 1.1 and HTTP 2 protocol for uplink. Usage of secure connections is possible using existing HTTP technologies.

The FLUS sink offers a simple HTTP PUT or POST interface for upload. The actual uplink stream is provided in the HTTP request body.

The FLUS sink exposes the *Push URL* element, which provides the base URL for the ingestion. All FLUS source appended sub-paths to the base URL belong to the same FLUS session.

Example, the FLUS sink offers the *Push URL* “http://sink.operator.com/sessionxyz/” via F-C. This allows the FLUS source to ingest sessions with multiple media components. Each media component is identified by a unique URL. The FLUS source appends additional path parts to complete the URL for the media. For example, the FLUS source sends audio to http://sink.operator.com/sessionxyz/audio-180130.mp4 and video to http://sink.operator.com/sessionxyz/video-180130.mp4.

When the FLUS source starts the media session, the FLUS source streams first the CMAF Header information for the movie file. After that, the FLUS source streams FLUS Chunks as the FLUS chunks become available.

In case of HTTP 1.1, the FLUS source uses HTTP chunked transfer encoding. Usage of HTTP chunked transfer encoding is indicated in the HTTP request header for the upload. The FLUS source finalizes the HTTP resource by sending a zero-size HTTP Chunk. HTTP 1.1 does not allow multiplexing of multiple simultaneous HTTP resources (aka media components), so, when multiple media components are streamed uplink, a separate TCP connection is needed for each media component.

In case of HTTP2, the FLUS source is simply omitting the Content-Length header. The FLUS source finalizes the HTTP resource by closing the HTTP2 stream using the END_STREAM flag in a frame. HTTP 2.0 allows multiplexing of multiple HTTP resources on the same transport connection (such as TCP).

When using TCP as transport, the usage of a persistent TCP connection for HTTP resource up streaming is recommended. The TCP buffer level is controlled by means of the TCP_NOTSENT_LOWAT socket option that is available in multiple operating systems.

An example of a recommended congestion control is LEDBAT [12], other congestion control schemes, which strive for a low network queue delay, are currently under development in IETF.

7.1.4.2 Rate Adaptation

The FLUS source could adapt the media bitrate to fit to the currently available link bitrate. The rate adaptation algorithms of the underlying transport protocol realization (such as TCP) are re-used. When the codec configuration parameters (e.g. picture parameter set) are not changed, the FLUS Source can change the encoding bitrate without interrupting the encoding process. A media streaming solution is preferably rate adaptive in order to cope with changing network conditions. A FLUS source, creating an fMP4 stream, can also change the bitrate as needed. In order to allow for rate adaptation, the FLUS sink offers a reception buffer (Cf. Figure 2), which delays the stream for a configurable duration. The reception buffer can be used to compensate link bitrate variations without the need of changing the media quality. The FLUS Source starts rate adaptation when the reception buffer on the FLUS Sink cannot compensate the link bitrate variations anymore. The reception buffer depth could be configurable via F-C.

The FLUS sink uses this reception queue (see figure below) to recover the encoder frame rate, i.e. to compensate network jitter. The FLUS source needs to know or needs to provision the FLUS sink delay in order to apply rate adaptation techniques for example to provide the best possible quality at minimal frame losses (i.e. due to late FLUS Sink arrival). Such a configuration is provided with the *Pipeline Description* element.

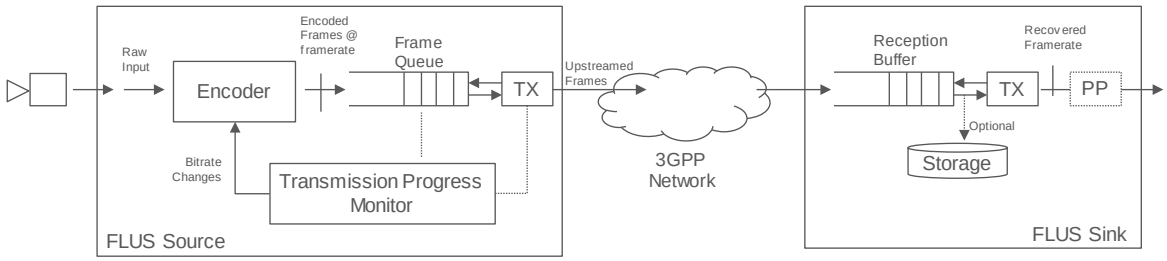


Figure 7.1.4.2-1: Rate Adaptation

A FLUS source can monitor the upstreaming progress or could listen to notifications / rate recommendations from the network. Either as an alternative and/or additional facility, the network could provide a dedicated assistance capability to boost the reception of upstream media, in case transient network throughput restrictions have caused a too high backlog at the FLUS Source. This model is particularly appropriate for approaches whereby the FLUS source expects the network to be able to upstream a particular pre-determined format or bitrate version of the media asset that is being sourced.

Existing transport protocols such as TCP employ a rate adaptation algorithm, which adjusts the TCP throughput to the available link bitrate. A rate adaptation logic can measure the bitrate at which the TCP sender is draining the frame queue. Further, the FLUS source can monitor, how quickly a frame is upstreaming (first byte of the frame until the last byte of the frame).

Any quality changes of the FLUS Source due to rate adaptation is noticeable to the audience. In some situations, it may be better to configure a larger Transmission Buffer (i.e. operate at a higher delay) to ensure a higher and sustainable

quality. There is no need to standardize the detailed rate adaptation algorithm. However, the FLUS sink should support a reception queue and recovery of the encoder frame rate.

Other transport protocols such as QUIC may also be used to re-use rate control and retransmission schemes.

7.1.5 fMP4-based Instantiation using multiple segments per track

7.1.5.1 General Description

This section contains a similar description to clause 7.1.2 with the difference, that the CMAF track is subdivided into individual CMAF Segments. Depending on the FLUS Sink implementation, every CMAF Segment is identified by a unique URL or the FLUS sink derives the segment sequence from the segments itself. In the first case, all CMAF Segments belonging to the same CMAF Track are identified by the same base URL. In the latter case, that same URL is used for all CMAF Segment of the same CMAF Track.

This instantiation is identified in the F-C configuration by the following urn: “org:3gpp:flus:2018:instantiations:fmp4”. An additional F-C configuration option determines, whether HTTP 1.1 (with Chunked Transfer Encoding), HTTP2.0 with TCP or HTTP 2.0 with other transport protocols such as QUIC should be used. Note, the HTTP version and the transport protocol may also be negotiated at connection setup. When using QUIC as transport protocol, the FLUS Source can fall back to TCP, when the QUIC session setup fails.

A FLUS session may contain one or more media components. When the FLUS session contains multiple media components, each component is formatted as a CMAF Track and upstreamed separately. A common presentation timeline is used across the different media components.

The CMAF Header contains information around the number of tracks, the used codec, codec configuration and optionally static metadata for the upstreamed movie file.

Every CMAF Track starts with the CMAF Header, followed by a sequence of CMAF Segments. Every CMAF Segment may contain one or more CMAF Chunks.

Every CMAF Segment starts with an ‘styp’ box.

An additional F-C configuration option allows the selection of the “Segmented” Profile. When selecting the “Segmented” Profile, some additional configuration parameters are needed. For every CMAF Track, an URL for the CMAF Header and an URL Template for CMAF segments is configured.

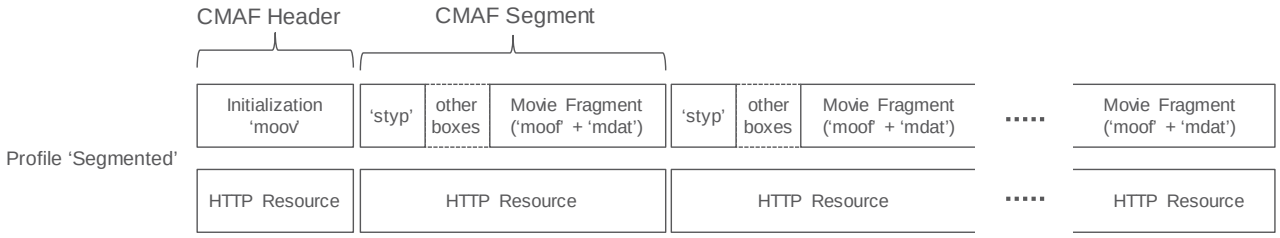


Figure 7.1.5.1-1: Illustration of Segmented Profiles with one CMAF Chunk per CMAF Segment (with mapping to HTTP resources)

The FLUS sink offers a simple HTTP PUT or POST interface for upload. The actual uplink stream is provided in the HTTP request body. The HTTP request header contains an URL, either corresponding to the CMAF header or is formatted according to the CMAF Segment URL template.

8 Example FLUS Workflows

8.1 Example Workflow using F-U MMTP

The UE uses the F-C RESTful procedures to create a FLUS session. It includes the identifier: “org:3gpp:flus:2018:instantiations:mmtp-ws”, to indicate that the requested F-U instantiation is fMP4 over

MMTP/WebSocket. If accepted by the FLUS Sink, it includes a websocket URL in the response as the entry point to the F-U. The UE uses the provided WebSocket URL to connect to the Sink and to start the F-U session.

In this example, the session consists of one audio stream (captured from a microphone) and 2 video streams (captured from two cameras). As such, 3 different MMTP sub-flows are created and assigned packet ids 1, 2, and 3. Each sub-stream carries a different component of the media data. At the start of the session, the MPT message is sent with an MPT table that describes the 3 different sub-flows as separate assets. The MMT_general_location_info indicates the associated packet_id to each asset using location_type=0x00.

To reduce the end-to-end latency, the Source creates CMAF Chunks that are of very short duration and sends them chunk by chunk. The CMAF chunk corresponds to an ISOBMFF fragment that is generated on the fly. The Source uses the same reference clock (NTP wall clock time) to synchronize the transmission of all sub-flows. The delivery timestamp is provided as part of every MMTP packet and reflects the wall clock time at the time of generation and transmission of the MMTP packet. Optimally, an MPU_timestamp_descriptor may also be generated and included as part of the MP table.

Figure 8.1-1 shows the structure of the F-U stream:

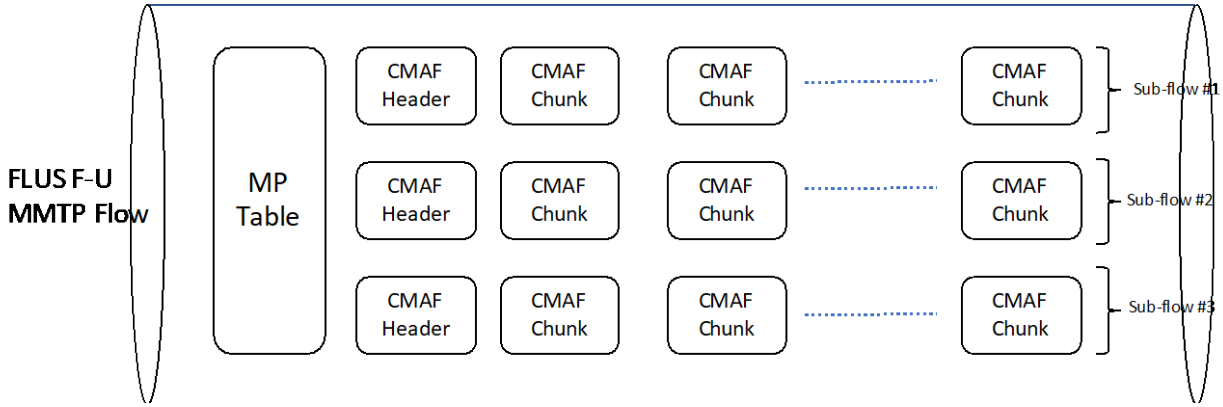


Figure 8.1-1: Structure of an MMTP flow as a FLUS F-U

Figure 8.1-2 depicts the message flow in the F-C to setup a FLUS session that uses the MMTP instantiation:

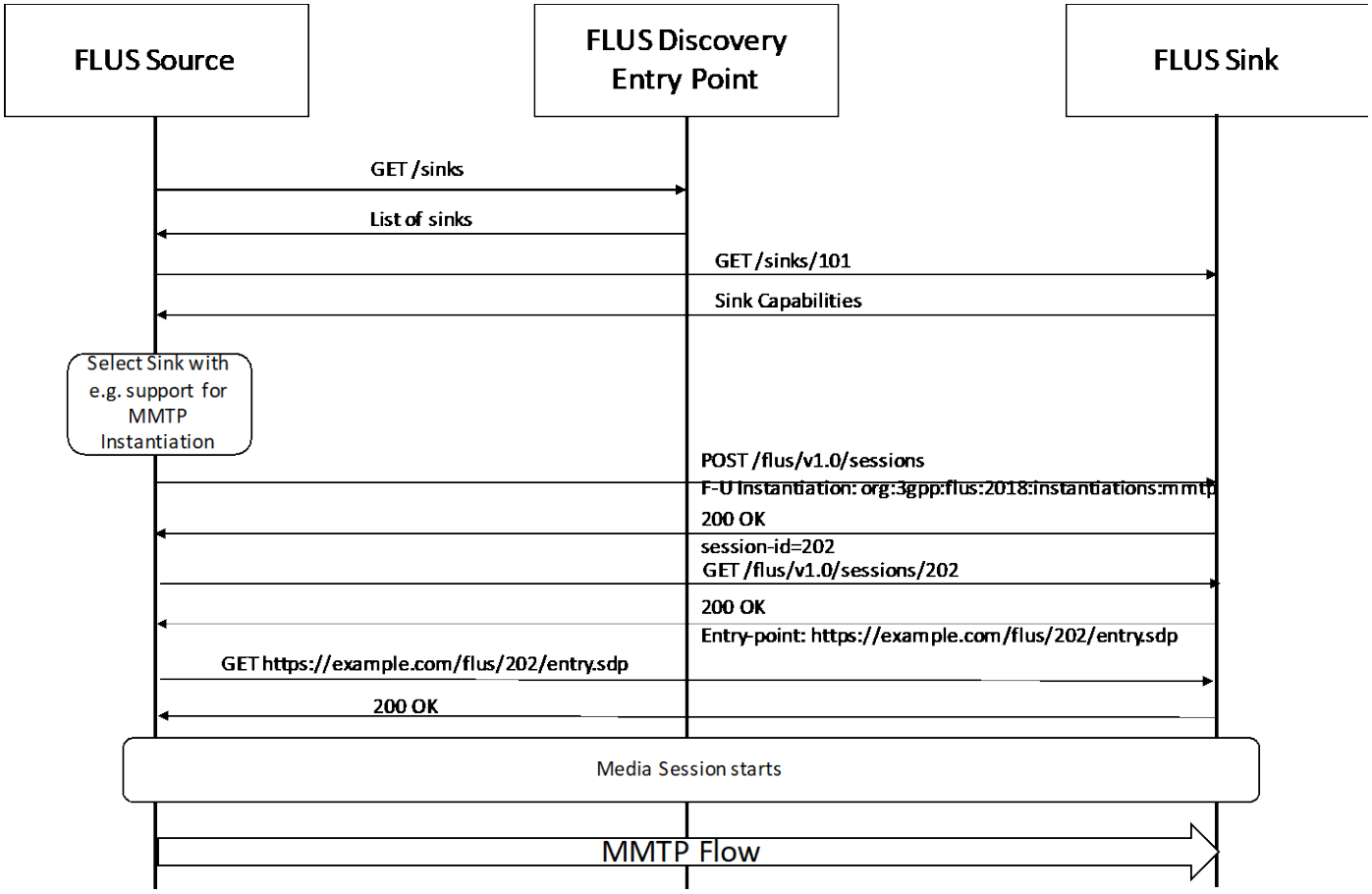


Figure 8.1-2: Message flow in the F-C to setup a FLUS session that uses the MMTP instantiation

After successful establishment of the FLUS session and fetching the SDP, the FLUS Sink will know the destination IP address and port number to which it will have to send the multiplexed MMTP Flow.

8.2 Example Call Flow for fragmented MP4 with HTTP Delivery

8.2.1 Assumptions

The FLUS session is created from the same device as the FLUS media plane is provided.

The FLUS Source selects a FLUS Sink, which supports the required Media Codecs and post processing capabilities.

The FLUS Sink provides a single HTTP Push URL so that the FLUS Source can establish one or more HTTP Sessions to the FLUS Sink. The Push URL here is http://sink.operator.com/sessionxyz/.

The FLUS Source uses the received Ingest URL and appends suffixes, so that any media component (CMAF Track) is identified by a unique URL. The FLUS Source here desires to upstream a video and an audio component.

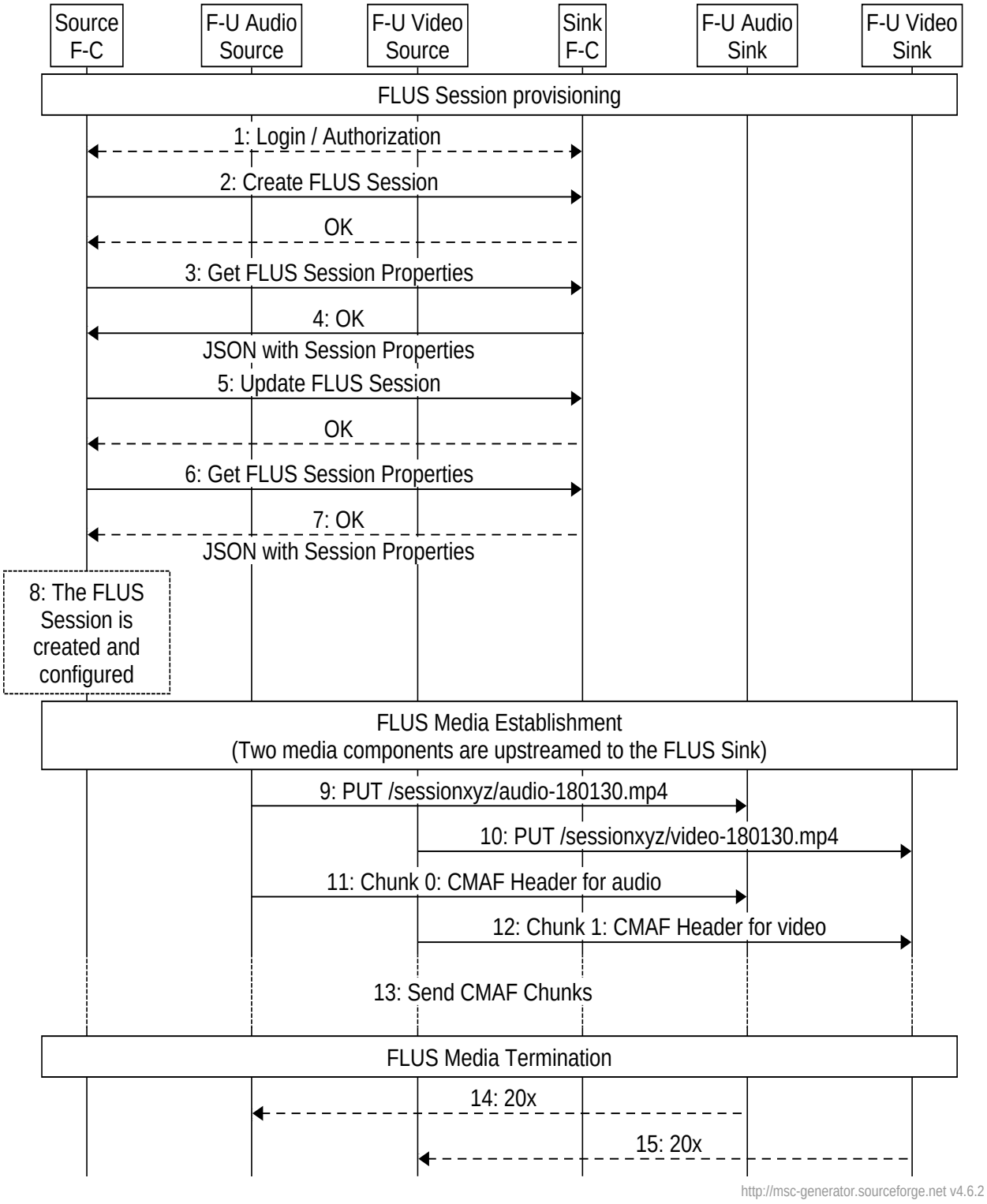


Figure 8.2.1-1: Example Call Flow for fragmented MP4 with HTTP Delivery

FLUS Session Provisioning using F-C

1) The User of the system (FLUS Source Control) logins into the FLUS system, e.g. using user-name and password.

2) The user creates a FLUS session using F-C. The FLUS Sink provides a unique session id to be used in subsequent transactions

3) For FLUS Session provisioning, the FLUS source first fetches the FLUS session parameters.

4) The FLUS Sink provides the FLUS session parameters with the response.

5) The FLUS Source does the needed FLUS Session modifications and applies the changes.

6) The FLUS source fetches the updated FLUS session parameters (The FLUS Sink may have done updates due to configurations).

7) The FLUS Sink provides the FLUS session parameters with the response. The FLUS Session parameters contains a Push base URL (here. http://sink.operator.com/sessionxyz/)

8) The FLUS Session is now fully provisioned and the FLUS Source has all needed information.

NOTE: The FLUS source function may be separated over different devices. A user can do the FLUS Session Configuration much earlier and potentially using a different device. When the FLUS Source re-connects to an already established, but not active session, the FLUS Source needs to authenticate towards the FLUS Sink (repetition of Step 1).

When time is due to establish the FLUS media session, the FLUS Source should establish here two HTTP Sessions for media components (each formatted as CMAF Track).

1) The FLUS Source establishes a transport connection (e.g. TCP) and sends an HTTP 1.1 Command. Here, the FLUS Source Sends an HTTP PUT command to establish the audio HTTP session.

2) The FLUS Source establishes a transport connection (e.g. TCP) and sends an HTTP 1.1 Command. Here, the FLUS Source Sends an HTTP PUT command to establish the video HTTP session.

3) The FLUS Source sends the CMAF Header file for the audio component. The FLUS sink finds detailed conduct configuration information.

4) The FLUS Source sends the CMAF Header file for the video component. The FLUS sink finds detailed conduct configuration information.

5) The FLUS Source starts appending CMAF Chunks to the established HTTP Sessions (according to the type),

When the FLUS Source is pausing the live uplink streaming session, the FLUS source may stop sending CMAF Chunk and keep the HTTP session open. When the FLUS session is continued, the FLUS Source may continue appending CMAF chunks to the session.

When time is due to terminate the FLUS media session, the FLUS Source sends a zero size HTTP Chunk.

6) Upon reception of the zero size HTTP Chunk, the FLUS Sink sends the HTTP response, indicating the creation of the audio track.

7) Upon reception of the zero size HTTP Chunk, the FLUS Sink sends the HTTP response, indicating the creation of the video track.

8.2.2 CMAF Format Example

An example from a wireshark capture is depicted below in Figure 8.2.2-1. The FLUS source uses here HTTP PUT together with HTTP chunked transfer encoding to an Apache2 server. The Apache2 server was configured with a webdav server module.

The first HTTP Chunk contains the ‘ftyp’box and the initialization information. The first HTTP chunk is of size ‘27d’. The first FLUS chunk (containing here only ‘moof’and ‘mdat’boxes) is set afterwards as single HTTP chunk. The size of the second HTTP chunk is 2beb.

```
PUT /webdav/dbg-DirCam-20180119-092131.mp4 HTTP/1.1
Transfer-Encoding: chunked
Content-Type: video/mp4
User-Agent: FLUS_HTTP_User_Agent
Host: 192.168.1.141
Connection: Keep-Alive
Accept-Encoding: gzip
Scheme: http

27d
....ftypisom....isomavc1...emoov...lmvhd.....[
..[
.....@.....trak...\tkhd.....[
..[
.....@.....emdia... mdhd.....[
..[
.....U.....%hdlr.....vide.....Tlos.....minf....vmhd.....$dinf....dref.....url
.....stbl....stsd.....|avc1.....H...H.....&avcC.B.
(....gB.(..@x....E8...h.C.....stts.....stsc.....stsz.....stco.....(mvex... Trex.....
.....2beb...Pmoof....mfhd.....8traf....tfhd...8.....d..+....@....trun.....X..+.mdat..+.e...@
...&(....}.....O.. 3..;.}.....]..}.....}..}.....?.G.W.Q.....'..x....>

<cut>
..Dns.@#.v'.....8..L#.....{..G.....?."8@F.....4F...B.B.'....7.#.C..8.<p.....8...D.0G...a.LG.#.x..>.>
...X`C.^...?8....?.P....
..<. mN#.....O?V..%..:.,#....q.z...V..b....U....7.%hK.xpC..."".....x....|Gb;..7
..ui..@.0Gb..#t!..w....Y..;.z..@!a..]Z...8LF.
0

HTTP/1.1 201 Created
Date: Fri, 19 Jan 2018 08:21:42 GMT
Server: KnownServer/2.4.18
Location: http://192.168.1.141/webdav/dbg-DirCam-20180119-092131.mp4
Access-Control-Allow-Origin: *
Content-Length: 291
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=ISO-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
```

```
<title>201 Created</title>
</head><body>
<h1>Created</h1>
<p>Resource /webdav/dbg-DirCam-20180119-092131.mp4 has been created.</p>
<hr />
<address>KnownServer/2.4.18 Server at 192.168.1.141 Port 80</address>
</body></html>
```

Figure 8.2.1-1: CMAF Format Example

When the FLUS source terminates the HTTP Request body using a zero size HTTP chunk, the HTTP server provides the HTTP response.

8.3 Example workflow for a drone mounted camera

8.3.1 Introduction

More and more cameras (and other media capturing devices) do not support a user interface for device provisioning, such as easily possible with today's SmartPhones. For example, a camera may be mounted to a drone for content capturing.

In case of the Framework for Live Uplink Streaming (FLUS), the functional architecture seem to assume that control plane (F-C) and user-plane (F-U) functions of the FLUS Source are always deployed together on the same physical device. Some instantiations may require the colocation of control and user plane functions. Other instantiations may support the separation of F-U and F-C.

8.3.2 Possible architecture

Figures 8.3.2-1a and 8.3.2-1b below depict possible architectures for the drone mounted camera use case. The arrows indicate the main information flows. The user plane part of the FLUS Source is deployed with the camera device. The control plane part for the FLUS Source resides in a separate device that is deployed with the administrator (typically a human operator) of the separate device, as denoted by "F-C" in either figure. An additional control plane part for remote control functionality, as denoted by "F-RC", is established between the FLUS Source and either a) the CTRL function residing in the separate device, as shown in Figure 8.3.2-1a, or b) the FLUS Sink, as shown in Figure 8.3.2.1b. Such remote-control-specific control plane functionality corresponds to remotely-provisioned control for the FLUS Source (e.g. starting or stopping the FLUS media, modifying encoding and capture settings). Additionally, there may exist a protocol interface between the control plane and the user plane parts of the FLUS Source, which is not in scope of FLUS.

Deployment options in addition to those shown in Figures 8.3.2-1a and 8.3.2-1b may be possible.

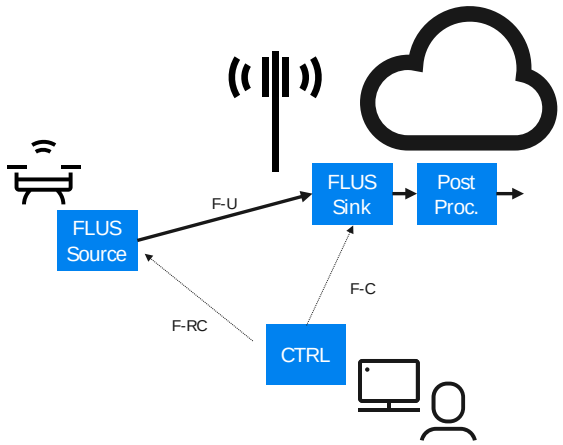


Figure 8.3.2-1a: Architecture Option 1 for Drone-mounted Camera - F-RC from CTRL to FLUS Source

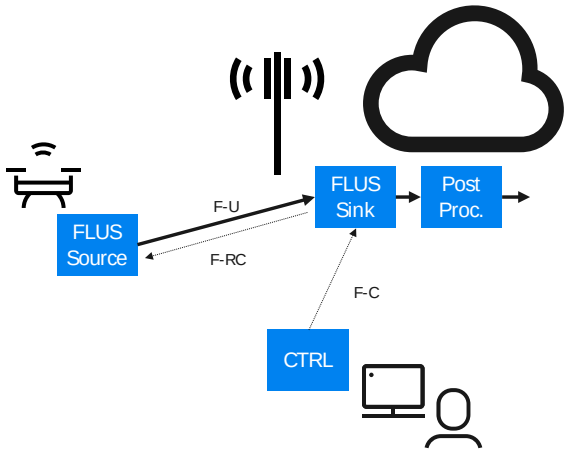


Figure 8.3.2.1b: Architecture Option 2 for Drone-Mounted Camera - F-RC from FLUS Sink to FLUS Source

The user plane part of the FLUS Source reads the data from the media capturing device, such as the video camera. The user plane function needs to have the following information provisioned by the CTRL function.

- Content Capturing parameters, such as capture device, coding configuration (encoding profile, bitrate, etc), upstream strategy (e.g., buffering).
- Content capturing start / stop times (schedule) or start / stop control mechanism.
- F-U parameters such as FLUS Sink address, used F-U protocol, and authentication credentials.

The CTRL function is responsible for the FLUS Sink configuration and the F-U instantiation selection and the post processing provisioning. The CTRL function also offers a User Interface so that the user can provision and monitor the set-up of the configuration, selection and provisioning functions.

In addition, a remote control session (F-RC) is established. The remote control session here is used to control the establishment, operation, and teardown of the FLUS media session.

8.3.3 Example Call Flow 1

The call flow in this clause illustrates an example according to Figure 8.3.2-1 (left), i.e. the remote control session (F-RC) is established between the Ctrl function and the drone (FLUS Source).

Here are example call flows for system provisioning and session activation to illustrate the information flow between the different functions.

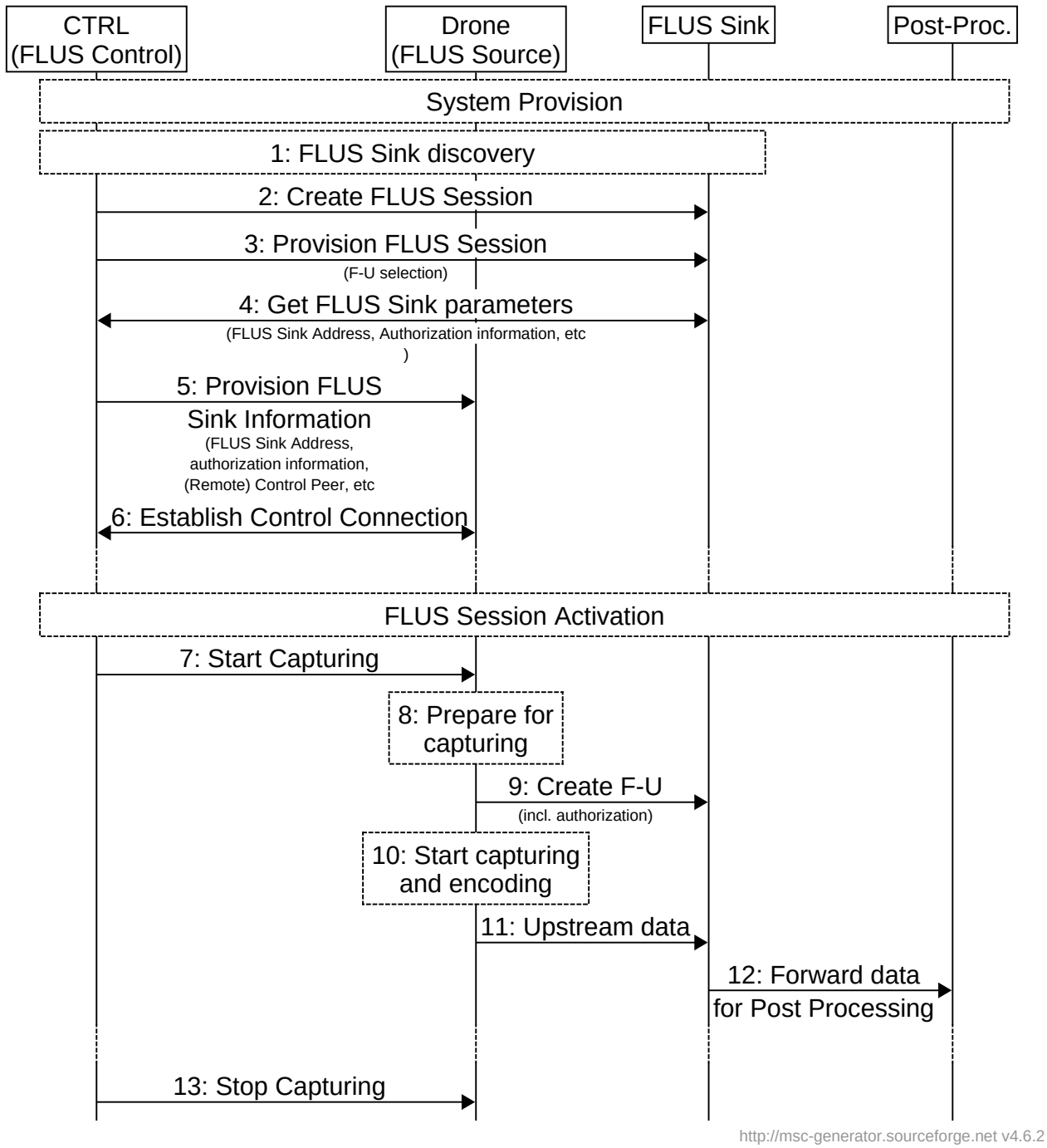


Figure 8.3.3-1: Example Call Flow 1

At time of System Provisioning

1: The FLUS Control (user of the system) needs to find a FLUS Sink with the appropriate capabilities. There are multiple ways to discover an appropriate FLUS Sink, e.g. using CAPIF or regular web offerings. Likely the most straight forward discovery procedure is to

- Go to the Web Offering of the one MNO and find the FLUS Sink catalogue
- Select the FQDN of an appropriate FLUS Sink
- Use your Browser or HTTP Rest Command line tool to connect to the FLUS Sink for FLUS Session provisioning

2: The FLUS Control (User) connects to the FLUS Sink for FLUS Session creation. The FLUS Control (User) may need to authenticate and authorize itself for FLUS Session Creation.

3: The FLUS Control (user) provisions the FLUS Session. That may include the following parameters

- F-U instance selection

- Provisioning of Post Processing (In the simplest case, the FLUS Session data are just stored in the Cloud)

4: The FLUS Control (user) gets FLUS Sink data. In particular F-U ingest entry information and authentication information, which is needed by the FLUS Source.

5: The FLUS Control (user) provisions the FLUS Source with information. The FLUS Control (user) provisions in particular the settings for capturing, encoding and upstreaming. The FLUS Control also provisions information for the FLUS Remote Control (F-RC) session. For upstreaming, the FLUS Sink ingest entry point information is provided together with authorization information.

6: The FLUS Source establishes the remote control session with the FLUS Control. The realization of the remote control session is ffs. This may be a WebSocket connection or another realization.

The FLUS system is now fully configured. The FLUS Source may activate the FLUS user plane session on remote control command.

At time of FLUS Session Activation

7: At scheduled FLUS media start up, the FLUS Control sends a remote control command to the FLUS Source to start the FLUS media session. Optionally, the FLUS Sink may provide the FLUS media start timestamp prior to the FLUS media start time.

8: At time of FLUS session activation, the FLUS Source starts preparing the capture process and configures the coding and upload procedures with the pre-provisioned parameters

9: The FLUS Source establishes the user plane to the FLUS Sink. The FLUS Source authenticates itself and provides the pre-provisioned authorization information (e.g. a token)

10: The FLUS Source starts the capturing and encoding procedures.

11: The FLUS Source starts upstreaming first video frames

12: The FLUS Sink forwards the received data to the post processing unit, which may simply store the frames onto a storage system

13: The FLUS Control may send a control command such as Stop to the FLUS Source. In this case, the FLUS Source stops the upload procedure. Other example commands might be pan, zoom or tilt to remote control the camera

8.3.4 Example Call Flow 2

The call flow in this clause illustrates an example according to Figure 8.3.2-1 (right), i.e. the remote control session (F-RC) is established between the FLUS Sink and the drone (FLUS Source).

Here are example call flows for system provisioning and session activation to illustrate the information flow between the different functions.

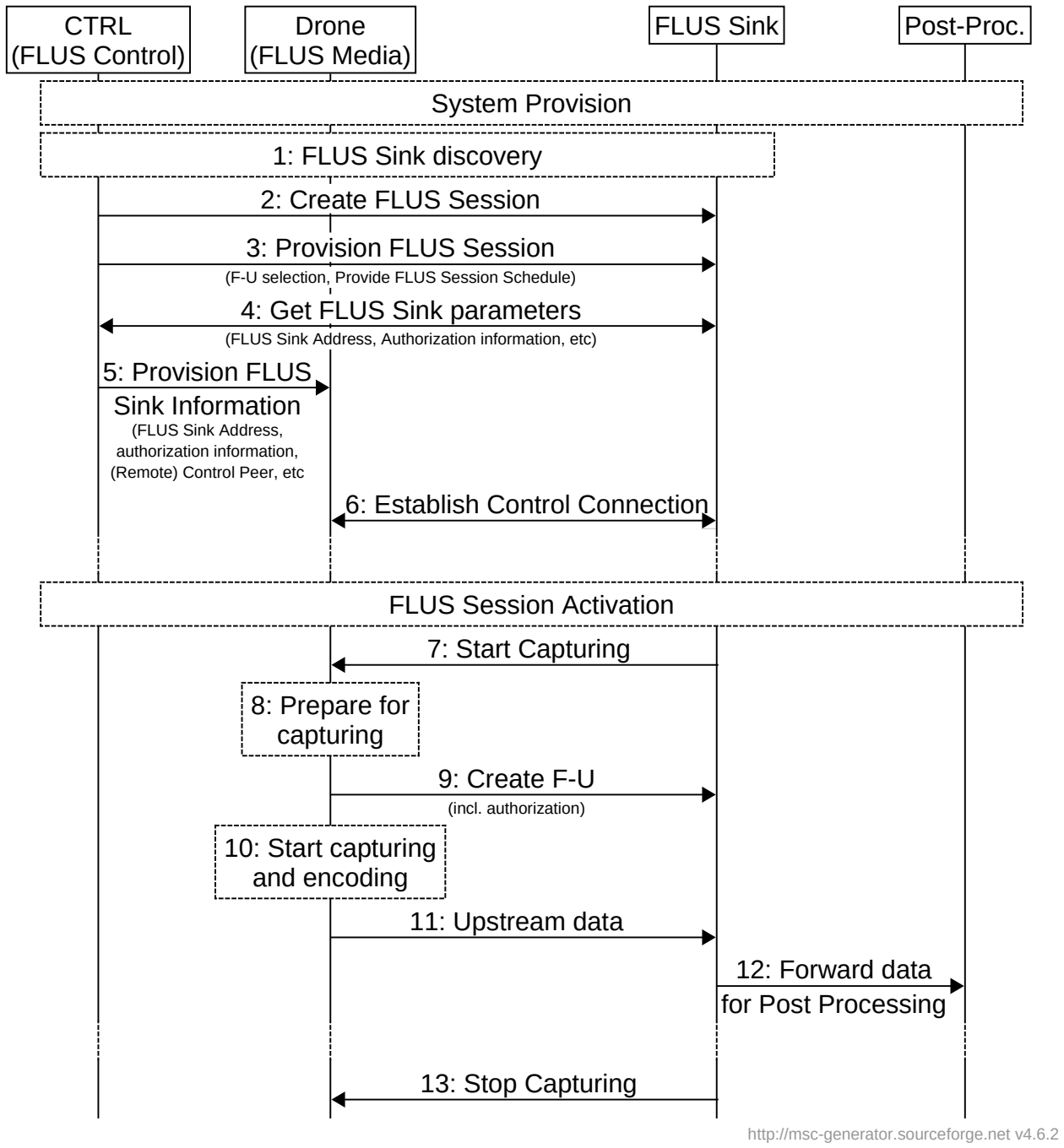


Figure 8.3.4-1: Example Call Flow 2

At time of System Provisioning

1: The FLUS Control (user of the system) needs to find a FLUS Sink with the appropriate capabilities. There are multiple ways to discover an appropriate FLUS sink, e.g. using CAPIF or regular web offerings. Likely the most straight forward discovery procedure is to

- Go to the Web Offering of the one MNO and find the FLUS Sink catalogue

- Select the FQDN of an appropriate FLUS Sink

- Use your Browser or HTTP Rest Command line tool to connect to the FLUS Sink for FLUS Session provisioning

2: The FLUS Control (User) connects to the FLUS Sink for FLUS Session creation. The FLUS Control (User) may need to authenticate and authorize itself for FLUS session creation.

3: The user provisions the FLUS session. That may include the following parameters

- F-U instance selection

- Provisioning of Post Processing. In the simplest case, the FLUS Session data are just stored in the Cloud.

- Start / Stop times (i.e. the specific times the FLUS sink is authorized to start / stop the FLUS session)

NOTE: This assumes that all the start/stop times are provided before FLUS media session activation.

4: The FLUS Control (user) retrieves FLUS Sink data. In particular F-U ingest entry information and authentication information of a FLUS Source User Plane function. The FLUS Session Schedule information is provided to the FLUS Sink with the provisioning step.

5: The FLUS Control (user) provisions the FLUS Source with information. The protocol is not in scope. The user provisions in particular the settings for capturing, encoding and upstreaming. For upstreaming, the FLUS Sink ingest entry point information is provided together with authorization information.

6: The FLUS Source establishes the remote control session with the FLUS Sink. The realization of the remote control session is ffs. This may be a WebSocket connection or another realization.

The FLUS system is now fully configured. The FLUS Source may activate the FLUS user plane session according to a schedule or on command.

At time of FLUS Session Activation

7: At scheduled FLUS media start up, the FLUS Sink sends a remote control command to the FLUS Source to start the FLUS media session. Optionally, the FLUS Sink may provide the FLUS media start timestamp prior to the FLUS media start time.

8: At time of FLUS Session Activation, the FLUS Source starts preparing the capture process and configures the coding and upload procedures with the pre-provisioned parameters

9: The FLUS Source establishes the user plane to the FLUS Sink. The FLUS Source authenticates itself and provides the pre-provisioned authorization information (e.g. a token)

10: The FLUS Source starts the capturing and encoding procedures.

11: The FLUS Source starts upstreaming first video frames

12: The FLUS Sink forwards the received data to the post processing unit, which may simply store the frames onto a storage system

13: The FLUS Sink may send a control command such as Stop to the FLUS Source. In this case, the FLUS Source stops the upload procedure. Other example commands might be pan, zoom or tilt to remote control the camera.

8. 4 Example FLUS call flow with Network Based Media Processing (NBMP)

8.4.1 Overview

In this subclause, various deployment scenarios of NBMP with FLUS are described. Each scenario consists of the deployment architecture, the call flows, and the employed interfaces. An overview of NBMP architecture can be found at [2] Annex B.

8.4.2 NBMP in the Application Server (All-AP)

8.4.2.1 Architecture:

This scenario is shown in Figure 8.4.2.1-1. In this case, NBMP Client, Workflow Manager, and MPEs are located in the Application Server.

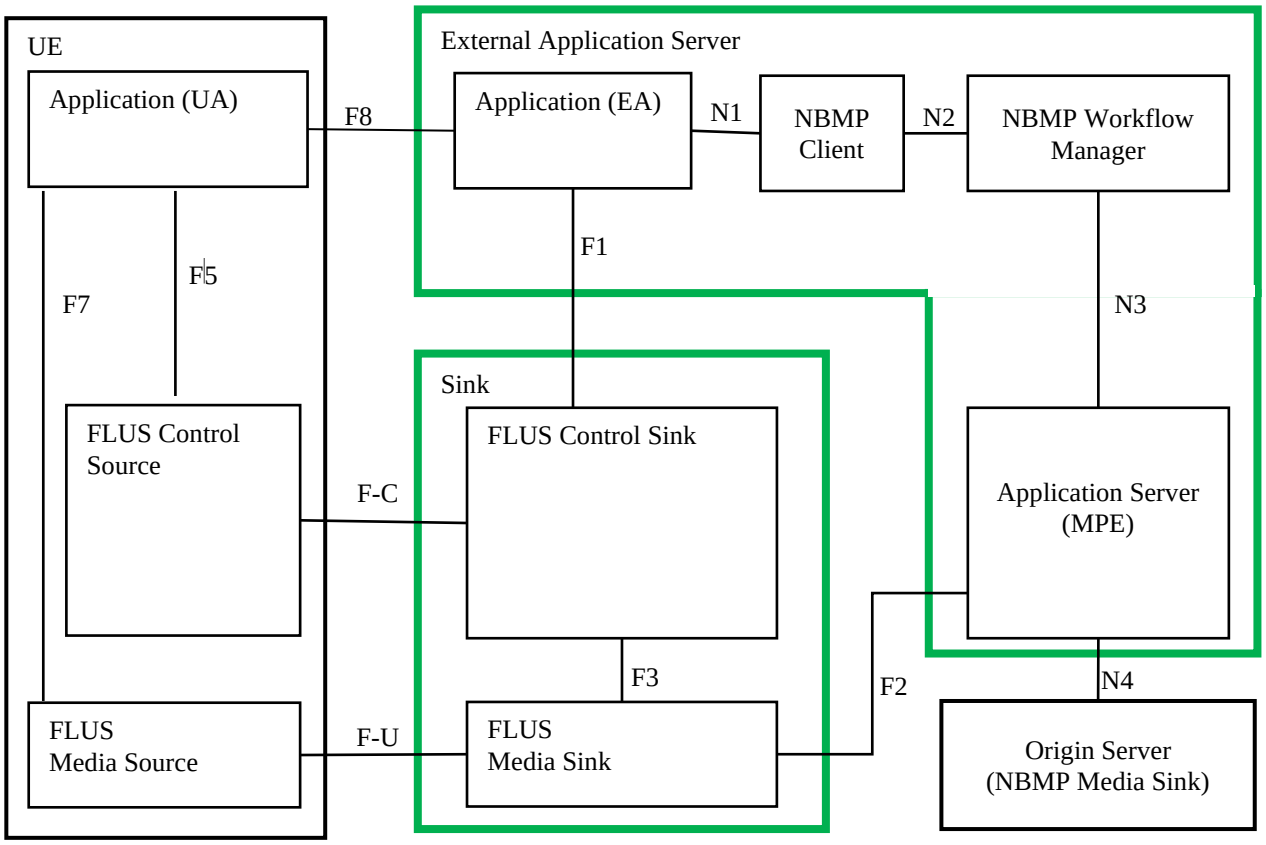
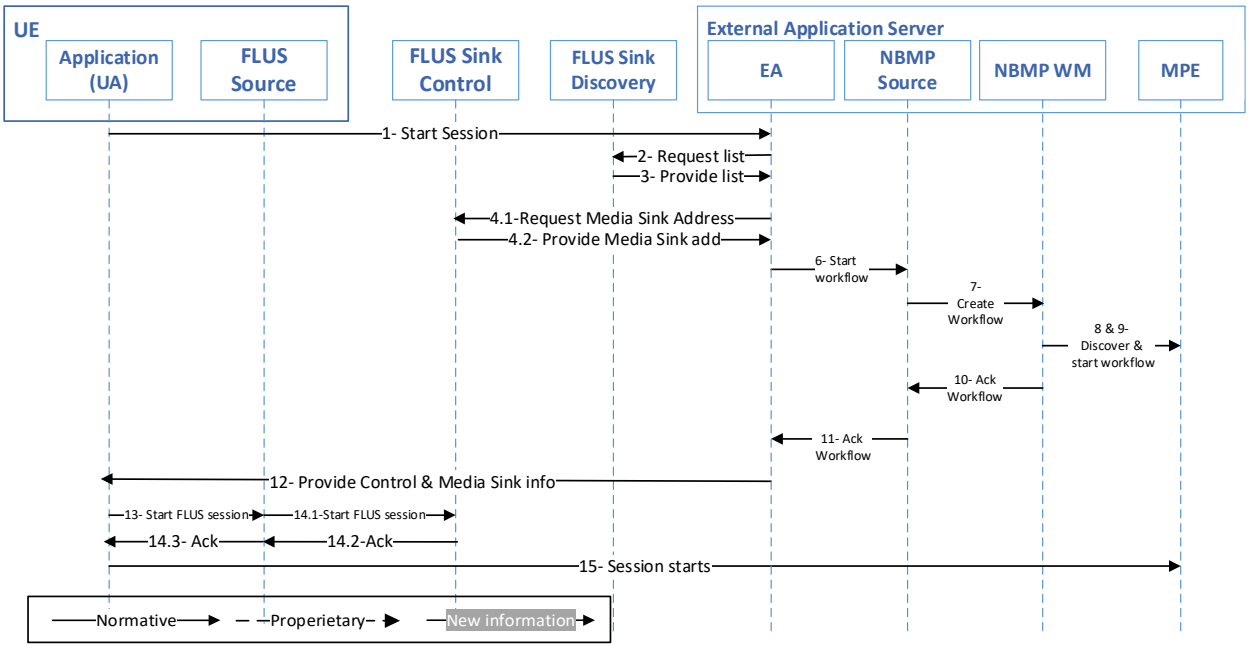


Figure 8.4.2.1-1: NBMP in Application Server

8.4.2.2 Call flow

The steps of establishing a FLUS-NBMP session are as the following



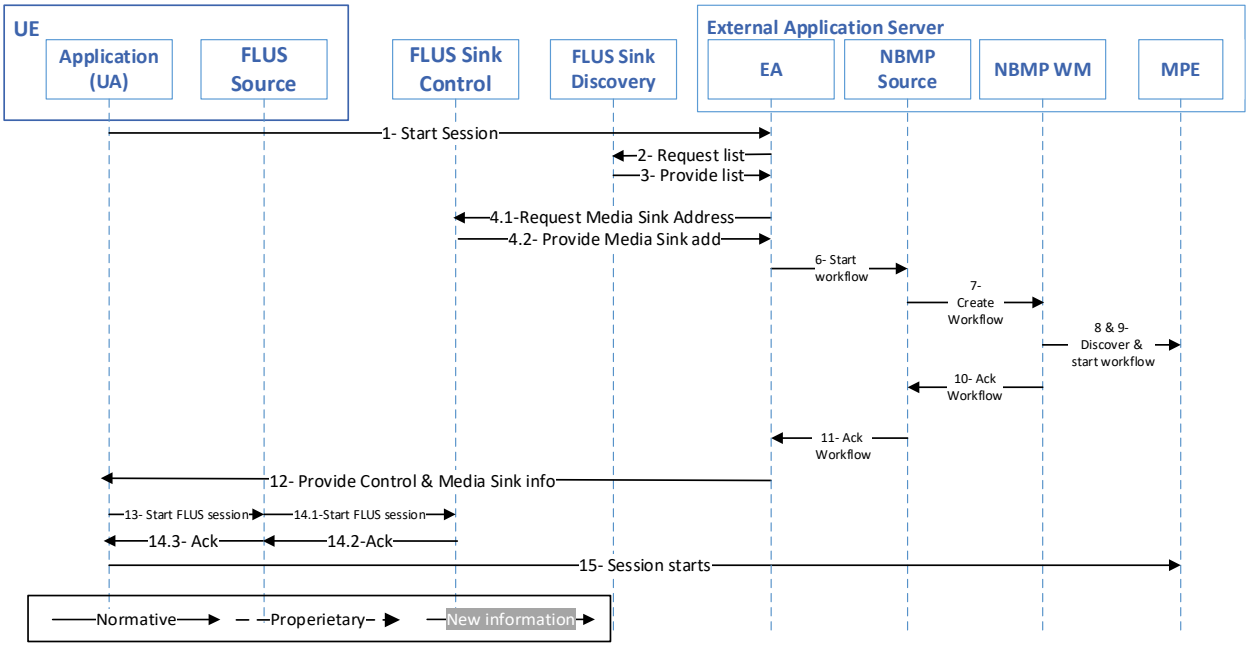


Figure 8.4.2.2-1: Call flow of NBMP in Application Server

The steps of establishing a FLUS-NBMP session are as the following:

1) UE Application (UA) makes a request through F8 to Application (EA) to start a live session.
2) EA requests the list of FLUS Sinks from a Sink Discovery Server.
3) Sink Discovery Server responds to EA's request.
4) EA picks a FLUS Sink and finds its FLUS Media Sink address.
5) EA retrieves the user profile and identifies the resources needed to run the service.
6) EA requests NBMP Client to start an NBMP Workflow.
7) NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow.
8) NBMP Workflow Manager discovers various MPEs and finds enough number of MPEs to run the workflow

   NOTE: SA6 discovery may be considered to address this functionality.

9) NBMP Workflow Manager instantiates the workflow.
10) NBMP Workflow responds to NBMP Client with updated WDD.
11) NBMP Client acknowledges workflow instantiation to EA.
12) EA responds to UA with Control Sink and Media Sink information.
13) UA requests FLUS Control Source to establish the FLUS session.
14) FLUS Control Source establishes the FLUS session and acknowledges UA.
15) The session starts.

8.4.2.3 Interfaces

Table 8.4.2.3-1 shows the required standard interfaces in this scenario:

Table 8.4.2.3-1: Required Standard APIs for NBMP in Application Server

Standard	FLUS	F-C, F-U, F1
	NBMP	N4, F2[1]

[1] The FLUS specification currently does not define setting up an output for FLUS Media Sink. To support this scenario, the FLUS specification needs to be extended to either support setting up an output address for FLUS Media Sink or provide an address for FLUS Media Sink's output for data retrieval.

NOTE: The internal APIs inside green boxes are out of the scope of this document.

8.4.2.4 Gap Analysis

This subclause provides a gap analysis for the above deployment scenario.

8.4.2.4.1 Mapping call flow to the standard APIs

The call flow presented in subclause 8.4.2.2 is mapped to the FLUS and NBMP APIs in the following table:

Table 8.4.2.4.1-1 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)
2. EA requests the list of FLUS Sinks from a Sink Discovery Server.	Supported by FLUS discovery API.
3. Sink Discovery Server responds to EA's request.	Supported by FLUS discovery API.
4. EA picks a Sink and finds its FLUS Media Sink address.	Not currently supported by FLUS.
5. EA retrieves the user profile and identifies the resources needed to run the service.	Out of scope (Internal to application).
6. EA requests NBMP Client to start an NBMP Workflow.	Out of scope (Internal to application).
7. NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow.	Supported by NBMP or External Application Provider specific.
8. NBMP Workflow Manager discovers various MPEs and finds enough number of MPEs to run the workflow NOTE: SA6 discovery may be considered to address this functionality.	Supported by NBMP spec.
9. NBMP Workflow Manager instantiates the workflow.	Supported by NBMP or External Application Provider specific.
10. NBMP Workflow responds to NBMP Client with updated WDD.	Supported by NBMP or External Application Provider specific.
11. NBMP Client acknowledges workflow instantiation to EA.	Supported by NBMP or External Application Provider specific.
12. EA responds to UA with Control Sink and Media Sink information.	Out of scope (Internal to application).

13. UA requests FLUS Control Source to establish the FLUS session.	Out of scope (Internal to application).
14. FLUS Control Source establishes the FLUS session and acknowledges UA.	Partially support by FLUS as indicated by 8.4.2.4.
15. The session starts.	Supported in FLUS and NBMP

8.4.2.4.2 TS 26.238 potential extensions

The FLUS specification currently does not define setting up an output for FLUS Media Sink. To support this scenario, the FLUS specification needs to be extended to either support setting up an output address for FLUS Media Sink or provide an address for FLUS Media Sink's output for data retrieval.

8.4.3 NBMP in the Application Server, MPE in Sink (MPE-Sink)

8.4.3.1 Architecture

This scenario is shown in Figure 8.4.3.1-1. In this case, NBMP Client and Workflow Manager are located in the Application Server, and MPEs are located in Sinks.

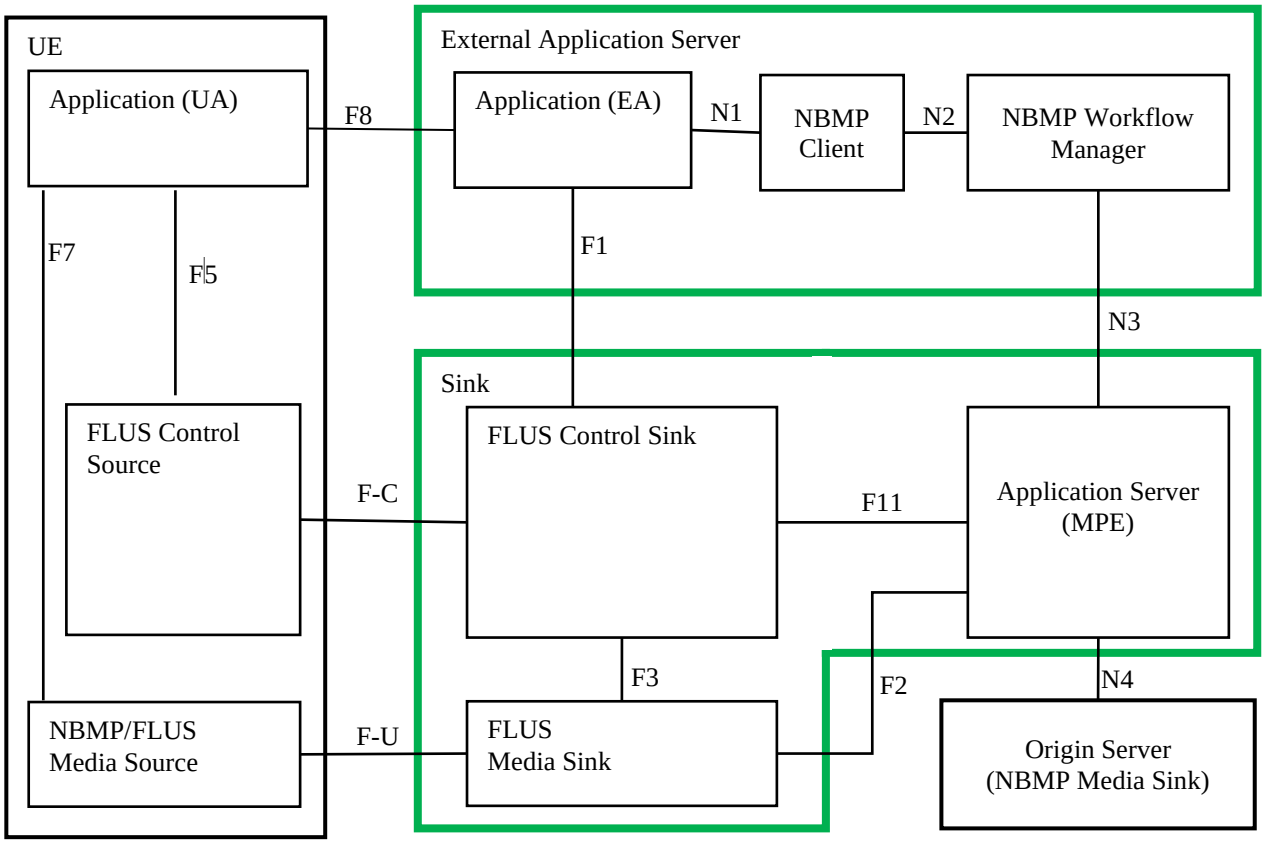


Figure 8.4.3.1-1: NBMP in Application Server, MPE in Sink

8.4.3.2 Call Flow

There are two possibilities of discovering MPE capabilities:

1) EA discovers MPE capabilities through FLUS Control Sink (F1).

2） EA discovers MPE's location through FLUS Control Sink （F1） and discovers the MPE capabilities through N3.

The call flows for both cases are shown below.

8.4.3.2.1 Through F1

The steps of establishing a FLUS-NBMP session are as the following:

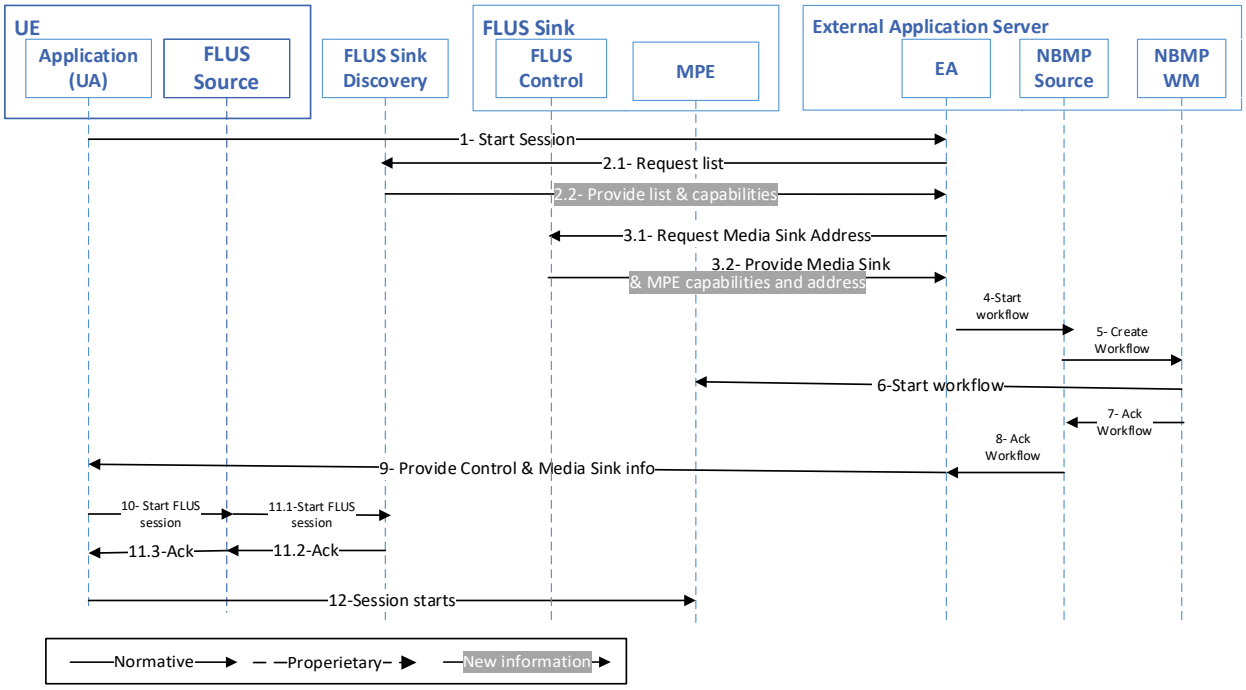


Figure 8.4.3.2.1-1: Call flow of NBMP in Application Server, MPE in Sink through F1

The steps of establishing a FLUS-NBMP session are as the following:

1） UE Application （UA） makes a request through F8 to Application （EA） to start a live session.
2） EA requests and receives the list of FLUS Sinks and their capabilities from the Sink Discovery Server.
3） EA picks a FLUS Sink that can run the workflow in its MPE and find its MPE address and MPE APIs in the Sink capabilities.
4） EA requests NBMP Client to start an NBMP Workflow with FLUS Media Sink Address.
5） NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.
6） NBMP Workflow Manager instantiates the workflow in the assigned MPE.
7） NBMP Workflow responds to NBMP Client with updated WDD.
8） NBMP Client acknowledges workflow instantiation to EA.
9） EA responds to UA with FLUS Control Sink and FLUS Media Sink information.
10） UA requests FLUS Control Source to establish the FLUS session.
11） FLUS Control Source establishes the FLUS session and acknowledges UA.
12） The session starts.

8.4.3.2.1 Through N3The steps of establishing a FLUS-NBMP session are as the following:

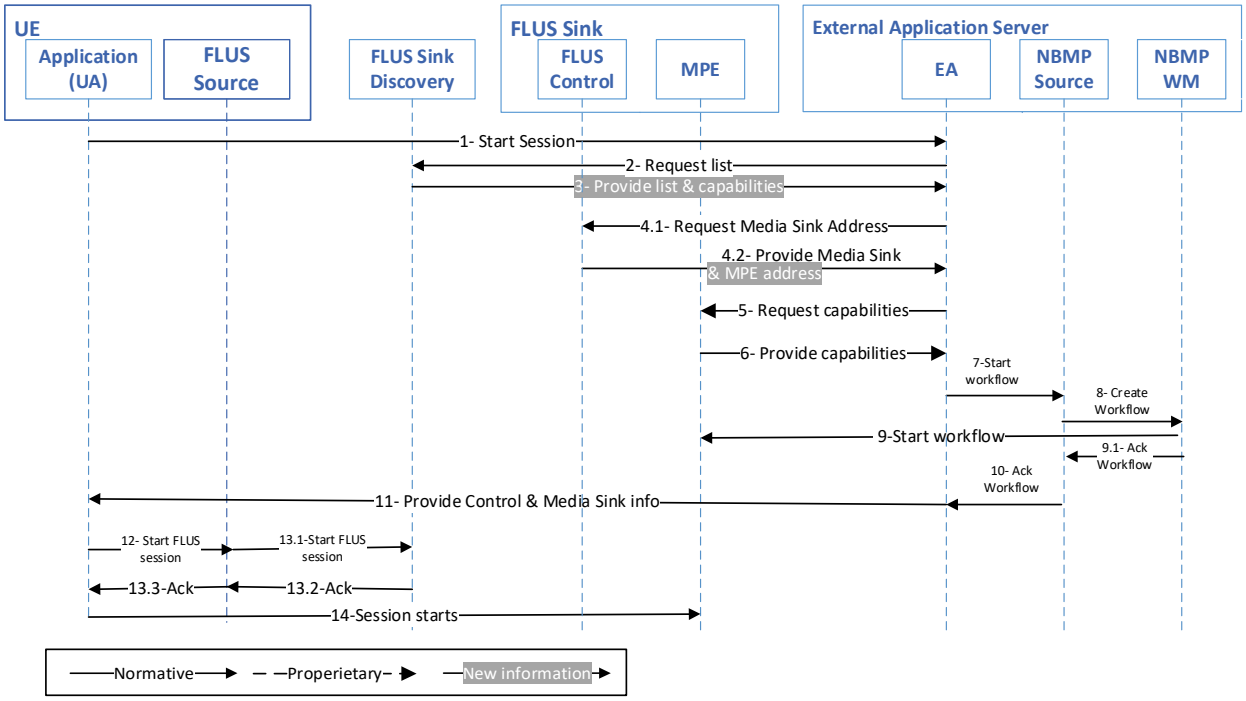


Figure 8.4.3.2.2-1: Call flow of NBMP in Application Server, MPE in Sink through N3

The steps of establishing a FLUS-NBMP session are as the following:

1) UE Application (UA) makes a request through F8 to Application (EA) to start a live session.
2) EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server.
3) The Sink Discovery Server provides the FLUS Sinks list and their capabilities.
4) EA picks a FLUS Sink that can run the workflow in its MPE and find its MPE address.
5) EA request MPE for its capabilities.
6) MPE provides its capabilities to EA.
7) EA requests NBMP Client to start an NBMP Workflow with FLUS Media Sink Address.
8) NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.
9) NBMP Workflow Manager instantiates the workflow in the assigned MPE and after establishing workflow, acknowledges to NBMP Client.
10) NBMP Client acknowledges workflow instantiation to EA.
11) EA responds to UA with Control Sink and Media Sink information.
12) UA requests FLUS Control Source to establish the FLUS session.
13) FLUS Control Source establishes the FLUS session and acknowledges UA.
14) The session starts.

8.4.3.2.1 Through N3

Table 8.4.3.3-1 shows the required standard interfaces in this scenario:

Table 8.4.3.3-1: Required Standard APIs for NBMP in Application Server, MPE in Sink

Standard	FLUS	F-C, F-U, F1
	NBMP	N4, N3[1]
[1] May be a closed API implemented by the Application Provider-operator agreement.		

NOTE: The internal APIs inside green boxes are out of the scope of this document

8.4.3.4 Gap analysis

This subclause provides a gap analysis for the above deployment scenarios.

8.4.3.4.1 Mapping call flow to the standard APIs

The call flow presented in clause 8.4.3.2.1 is mapped to the FLUS and NBMP APIs in the following table:

Table 8.4.3.4.1-1 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)
2. EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server.	Supported by FLUS discovery API.
3. EA picks a FLUS Sink that can run the workflow in its MPE and find its MPE address and MPE APIs in its capabilities.	Partially supported by FLUS. The EA discovers locations and optionally the capabilities of each FLUS Sink. The FLUS Sink can list the NBMP MPE identifier (URI) and optionally the MPE capabilities description in the Sink capabilities, but the inclusion of the URL address of the MPE is not currently supported by FLUS.
4. EA requests NBMP Client to start an NBMP Workflow with the provided FLUS Media Sink Address.	Out of scope (Internal to application).
5. NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.	Supported by NBMP or External Application Provider specific.
6. NBMP Workflow Manager instantiates the workflow in the assigned MPE.	Supported by NBMP spec/ the exact API is MNO specific.
7. NBMP Workflow Manager responds to NBMP Client with updated WDD.	Supported by NBMP spec.
8. NBMP Client acknowledges workflow instantiation to EA.	Out of scope (Internal to application).
9. EA responds to UA with FLUS Control Sink and FLUS Media Sink information.	Out of scope (Internal to application).
10. UA requests FLUS Control Source to establish the FLUS session.	Out of scope (Internal to application).
11. FLUS Control Source establishes the FLUS session and acknowledges UA.	Supported by FLUS
12. The session starts.	Supported in FLUS and NBMP

The call flow presented in clause 8.4.3.2.2 is mapped to the FLUS and NBMP APIs in the following table:

Table 8.4.3.4.1-2 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)
2. EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server.	Supported by FLUS discovery API.
3. The Sink Discovery Server provides the FLUS Sinks list and their capabilities.	Supported by FLUS discovery API.
4. EA picks a Sink that can run the workflow in its MPE and find its MPE address.	Partially supported by FLUS. The EA discovers locations and optionally the capabilities of each FLUS Sink. The FLUS Sink can list the NBMP MPE identifier (URI) and optionally the MPE capabilities description in the Sink capabilities, but the inclusion of the URL address of the MPE is not currently supported by FLUS.
5. EA request MPE for its capabilities.	Supported by NBMP.
6. MPE provides its capabilities to EA.	Supported by NBMP.
7. EA requests NBMP Client to start an NBMP Workflow with the provided FLUS Media Sink Address.	Out of scope (Internal to application).
8. NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.	Supported by NBMP spec or External Application Provider specific.
9. NBMP Workflow Manager instantiates the workflow in the assigned MPE and after establishing workflow, acknowledges to NBMP Client.	Supported by NBMP spec/ the exact API is MNO specific.
10. NBMP Client acknowledges workflow instantiation to EA.	Out of scope (Internal to application).
11. EA responds to UA with FLUS Control Sink and FLUS Media Sink information.	Out of scope (Internal to application).
12. UA requests FLUS Control Source to establish the FLUS session.	Supported by FLUS
13. FLUS Control Source establishes the FLUS session and acknowledges UA.	Supported by FLUS.
14. The session starts.	Supported in FLUS and NBMP.

8.3.4.3.2 TS 26.238 potential extensions

As is shown in clause 8.4.3.4.1, only steps 3 and 4 of Tables 8.4.3.4.1-1 and 8.4.3.4.1-2 respectively are not fully supported by TS 26.238. The capability signaling of FLUS Sink is defined by TS 26.238 Table 7.1.1.1-1. In this table,

the support of the NMBP can be signalled using the “scheme” item. The optional “location” provides the description for the capability to be retrieved. However, to directly access the NBMP Workflow Manager (WM), the location of WM is needed to be signaled. Therefore, we need to either:

1) Add the URL location of WM to the description
2) Add another item in the capabilities array item for the actual address

An Example for adding the support for NBMP Workflow Manager, with a description of NBMP Workflow Manager:

{ “scheme” : “urn:mpeg:mpegi:nbmp:workflowmanager: 2020”,
“location”: “http://vnd.com/xzy/nbmpwm_description.json”
}

The document nbmpwm_description contains the description information about the WM. Currently, the ISO/IEC 23090-8 doesn’t define a description for WM. The MPEG NBMP CDAM2 defines a description of MPE, i.e. MPE capabilities. Therefore, either

1) Solution 1: a standard object for NBMP Workflow description can be created that has the WM address such as:

{

“url”: “http://10.20.30.40”,

“name”: “vnd-workflow-manager-1”,

“description”: “the workflow manager provided by VND, version 1, 2021”,

…..

}

2) Or, solution 2: a new item can be added in the capabilities:

{ “scheme” : “urn:mpeg:mpegi:nbmp:workflowmanager: 2020”,
“location”: “http://vnd.com/xzy/nbmpwm_description.json”,
“url”: “http://10.20.30.40”

}

8.4.4 NBMP Client in the Application Server, NBMP Workflow Manager, and MPE in Sink (WM-MPE-Sink)

8.4.4.1 Architecture

This scenario is shown in Figure 8.4.4.1-1.

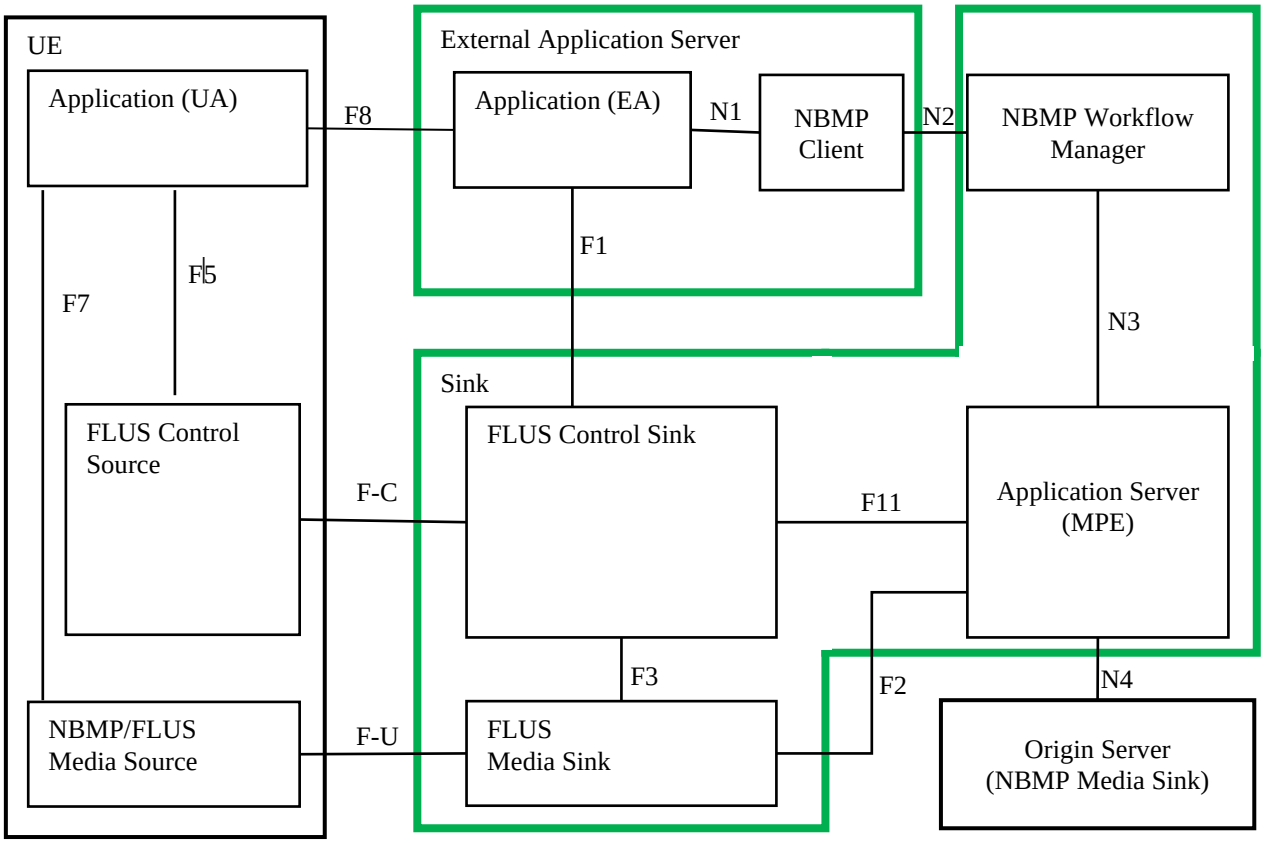


Figure 8.4.4.1-1: NBMP Client in Application Server, NBMP Workflow Manager, and MPE in Sink

8.4.4.2 Call Flow

The steps of establishing a FLUS-NBMP session are as the following:

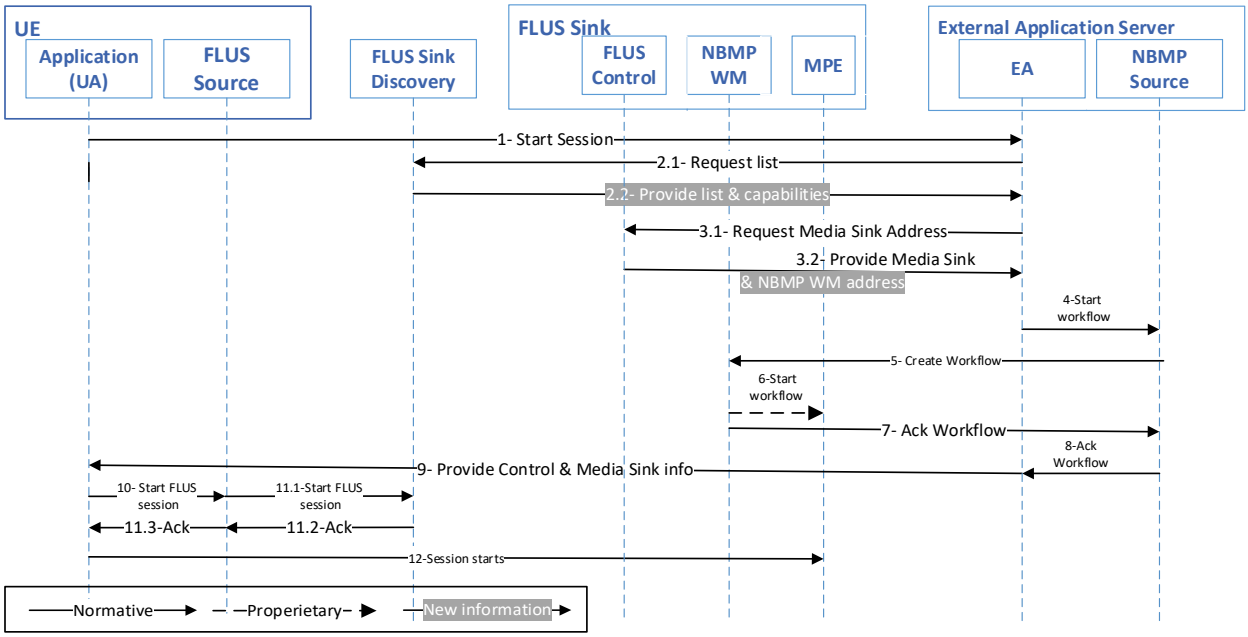


Figure 8.4.4.2-1: Call flow for NBMP Client in Application Server, NBMP Workflow Manager, and MPE in Sink

Same variations (discovering the entire MPE capabilities through FLUS Control Sink vs discovering MPE location through FLUS Control Sink) are possible here.

The steps of establishing a FLUS-NBMP session are as the following:

1) UE Application (UA) makes a request through F8 to Application (EA) to start a live session.
2) EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server.
3) EA picks a FLUS Sink that can run the workflow in its MPE and find its NBMP Workflow Manager and Media Sink address in the Sink capabilities.
4) EA requests NBMP Client to start an NBMP Workflow with FLUS Media Sink Address.
5) NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.
6) NBMP Workflow Manager instantiates the workflow in the assigned MPE.
7) NBMP Workflow responds to NBMP Client with updated WDD.
8) NBMP Client acknowledges workflow instantiation to EA.
9) EA responds to UA with Control Sink and Media Sink information.
10) UA requests FLUS Control Source to establish the FLUS session
11) FLUS Control Source establishes the FLUS session and acknowledges UA
12) The session starts.

8.4.4.3 Interfaces

Table 8.4.4.3-1 shows the required standard interfaces in this scenario.

Table 8.4.4.3-1: NBMP Client in Application Server, NBMP Workflow Manager, and MPE in Sink

Standard	FLUS	F-C, F-U, F1
	NBMP	N2, N4

NOTE: The internal APIs inside green boxes are out of the scope of this document.

8.4.4.4 Gap Analysis

This subclause provides a gap analysis for the above deployment scenario.

8.4.4.5 Mapping call flow to the standard APIs

The call flow presented in subclause 8.4.4.2 is mapped to the FLUS and NBMP APIs in the following table:

Table 8.4.4.4.1-1 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)
2. EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server (not shown).	Supported by FLUS discovery API from the FLUS discovery server.

3. EA picks a FLUS Sink that can run the workflow in its MPE and find its NBMP Workflow Manager and Media Sink address in the Sink capabilities.	Partially supported by FLUS. The EA discovers locations and optionally the capabilities of each FLUS Sink. The FLUS Sink can list the NBMP MPE identifier (URI) and optionally the MPE capabilities description in the Sink capabilities, but the inclusion of the URL address of the NBMP Workflow Manager is not currently supported by FLUS.
4. EA requests NBMP Client to start an NBMP Workflow with the provided FLUS Media Sink Address.	Out of scope (Internal to application).
5. NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.	Supported by NBMP or External Application Provider specific.
6. NBMP Workflow Manager instantiates the workflow in the assigned MPE.	Supported by NBMP spec/ the exact API is MNO specific.
7. NBMP Workflow responds to NBMP Client with updated WDD.	Supported by NBMP spec.
8. NBMP Client acknowledges workflow instantiation to EA.	Out of scope (Internal to application).
9. EA responds to UA with FLUS Control Sink and FLUS Media Sink information.	Out of scope (Internal to application).
10. UA requests FLUS Control Source to establish the FLUS session	Out of scope (Internal to application).
11. FLUS Control Source establishes the FLUS session and acknowledges UA	Supported by FLUS.
12. The session starts.	Supported in FLUS and NBMP.

8.4.4.4.2 TS 26.238 potential extensions

As is shown in 8.4.4.4.1, the FLUS specification can be extended to support the discovery of the NBMP Workflow Manager's address. Subclause 8.4.3.4.2 addresses this feature.

8.4.5 NBMP Client in the FLUS Control Source, NBMP Workflow Manager, and MPE in Sink (NBMPSource-FLUSSource)

8.4.5.1 Architecture

This scenario is shown in Figure 8.4.5.1-1.

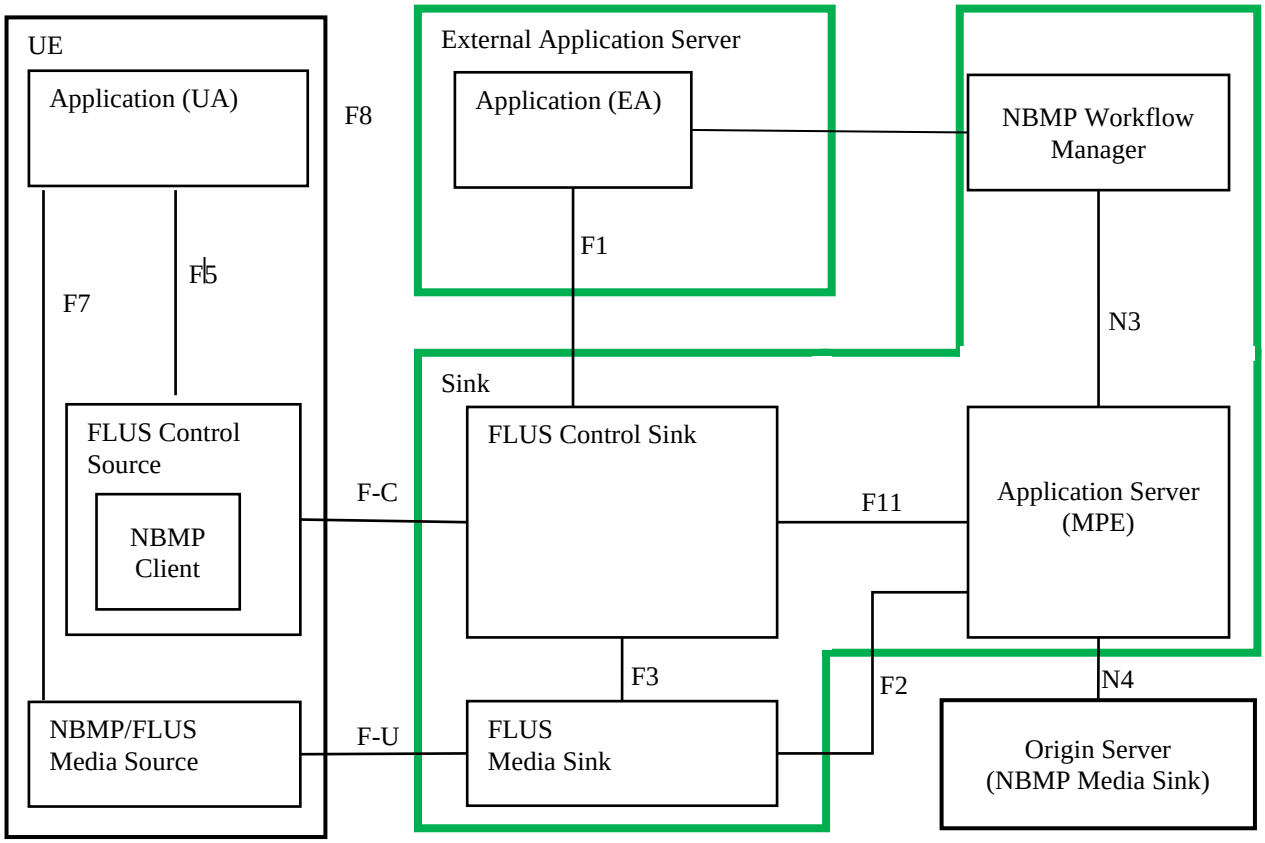


Figure 8.4.5.1-1: NBMP Client in FLUS Control Source, NBMP Workflow Manager, and MPE in Sink

8.4.5.2 Call Flow

The steps of establishing a FLUS-NBMP session are as the following:

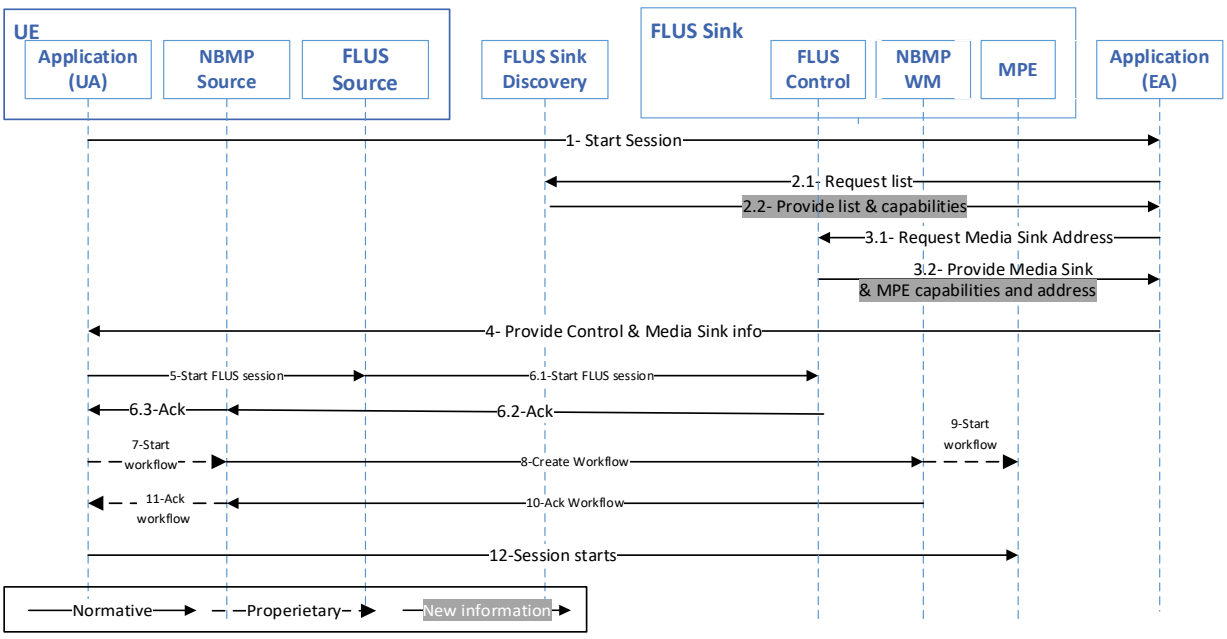


Figure 8.4.5.2-1: Call flow for NBMP Client in FLUS Control Source, NBMP Workflow Manager, and MPE in Sink

In this case, the UE can either

1) provide NBMP Workflow's location, so that NBMP Client can discover the MPE capabilities through that API as well as establishing the workflow, or
2) discover capabilities of Sink during the sink discovery process, so that NBMP Client doesn't need to discover capabilities of a sink's MPE in a different step.

The steps of establishing and operating a FLUS-NBMP session are as the following:

1) UE Application (UA) makes a request through F8 to Application (EA) to start a live session.
2) EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server.
3) EA picks a Sink that can run the workflow in its MPE and find its NBMP Workflow Manager and Media Sink address in the Sink capabilities.
4) EA responds to UE with the NBMP Workflow Manager's full URL or a relative URL through FLUS Control Sink.
5) UA requests FLUS Control Source to establish the FLUS session.
6) FLUS Control Source establishes the FLUS session and acknowledges UA.
7) UA requests NBMP Client to start the workflow.
8) NBMP Client builds WDD, and requests NBMP Workflow Manager (directly or through FLUS Control Sink) to instantiate the Workflow.
   Note: The details of this step are not shown in the call flow diagram.
9) NBMP Workflow Manager instantiates the workflow in the MPE.
10) NBMP Workflow responds to NBMP Client with updated WDD.
11) NBMP Client acknowledges workflow instantiation to UA.

The session runs.

8.4.5.3 Interfaces

Table 8.4.5.3-1 shows the required standard interfaces in this scenario:

Table 8.4.5.3-1: NBMP Client in FLUS Control Source, NBMP Workflow Manager, and MPE in Sink

Standard	FLUS	F-C[1], F-U, F1
	NBMP	N4
[1] With the support of NBMP Workflow Manager APIs		

NOTE: The internal APIs inside green boxes are out of the scope of this document.

8.4.5.4 Gap Analysis

This subclause provides a gap analysis for the above deployment scenario.

8.4.5.4.1 Mapping call flow to the standard APIs

The call flow presented in clause 8.4.5.2 is mapped to the FLUS and NBMP APIs in the following table:

Table 8.4.5.4.1-1 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)

2. EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server (not shown).	Supported by FLUS discovery API from the FLUS discovery server.
3. EA picks a FLUS Sink that can run the workflow in its MPE and find its NBMP Workflow Manager and Media Sink address in the Sink capabilities.	Partially supported by FLUS. The EA discovers locations and optionally the capabilities of each FLUS Sink. The FLUS Sink can list the NBMP MPE identifier (URI) and optionally the MPE capabilities description in the Sink capabilities, but the inclusion of the URL address of the MPE is not currently supported by FLUS.
4. EA responds to UE with the NBMP Workflow Manager's full URL or a relative URL through FLUS Control Sink.	Out of scope (Internal to application).
5. UA requests FLUS Control Source to establish the FLUS session	Out of scope (Internal to application).
6. FLUS Control Source establishes the FLUS session and acknowledges UA	Supported by FLUS spec.
7. EA requests NBMP Client start the workflow.	Out of scope (Internal to application).
8. NBMP Client builds WDD, and requests NBMP Workflow Manager (directly or through FLUS Control Sink) to instantiate the Workflow	Directly: Supported by NBMP Spec Through FLUS sink: Supported by FLUS spec
9. NBMP Workflow Manager instantiates the workflow in the MPE.	Out of scope (Internal to application).
10. NBMP Workflow responds to NBMP Client with updated WDD.	Directly: Supported by NBMP Spec Through FLUS sink: Partially supported by FLUS Spec.
11. NBMP Client acknowledges workflow instantiation to UA.	Out of scope (Internal to application).
12. The session starts.	Supported in FLUS and NBMP.

8.4.5.4.2 TS 26.238 potential extensions

The sink capability discovery of TS 26.238 is adequate for the discovery of NBMP WM support by FLUS Sink.

The communication of the WDD is achieved through the Sink resource which is defined by TS 26.238 Table 5.3.6-1. In this table, the 'url' item provides the location of NBMP WM. Therefore, the NBMP Client can interact with NBMP WM directly. However, if NBMP Client is expected to interact with NBMP WM through the FLUS F-C link, then sending WDD and receiving it possible in the NBMP Workflow API's synchronous mode only. The asynchronous mode may suboptimally work if the response includes the NBMP WM URL, which means that the retrieval of the WDD is direct. Also, the wait time for retrieving the WDD is not provided in this case.

For complete support of NBMP Workflow API asynchronous mode, the Sink configuration needs to be extended to include HTTP headers in the response in addition to the already included resource (WDD

8.4.6 NBMP Client in the UA, NBMP Workflow Manager in the Application Server, and MPE in Sink (NBMPSourceUA-NBMPWMAS)

8.4.6.1 Architecture

In this scenario as shown in the following figure, NBMP Client is in the UA, NBMP WM is located in the Application Server and MPE is in Sink.

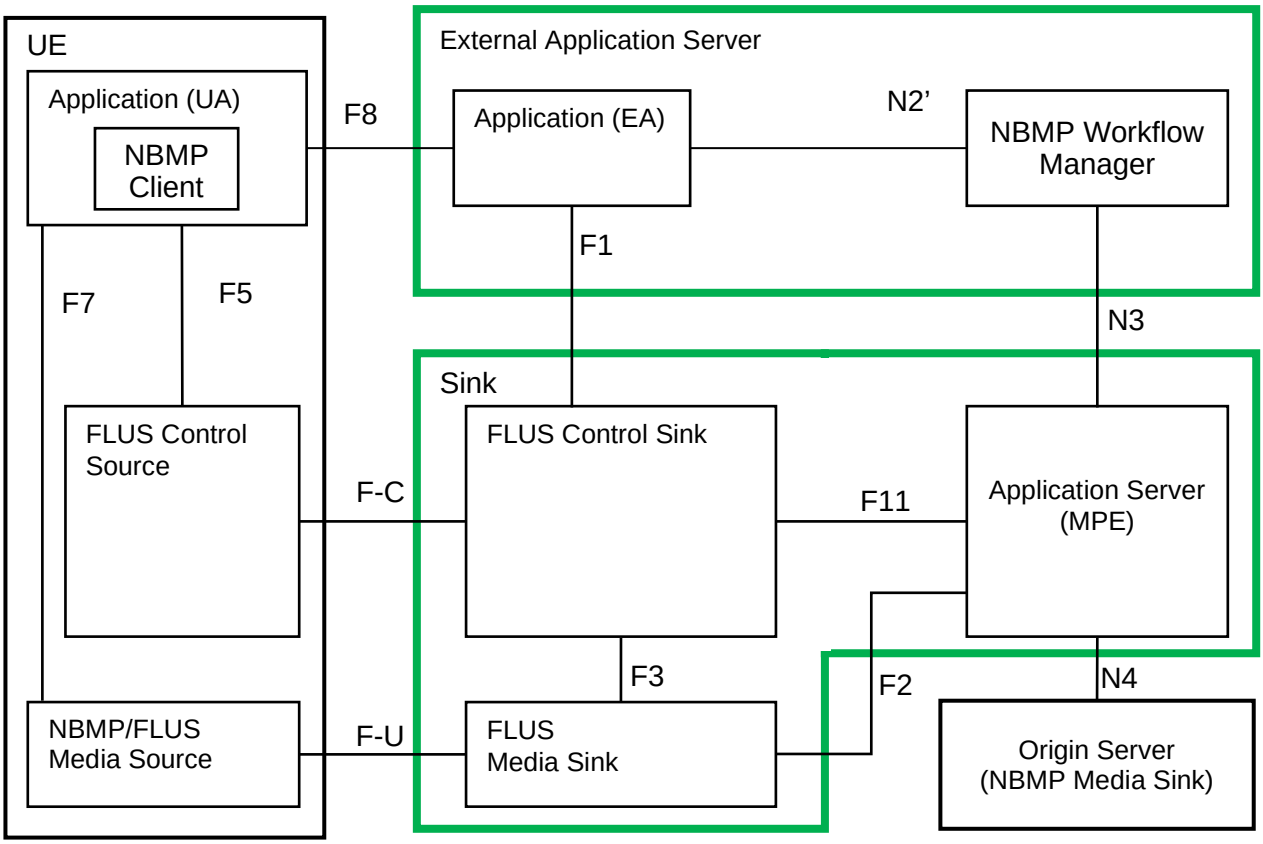


Figure 8.4.6.1-1: NBMP Client in the UA, NBMP Workflow Manager in the Application Server, and MPE in Sink

8.4.6.2 Call flow

This scenario is similar to the case when NBMP Client and NBMP WM are in the Application Server and MPE is in Sink as described in clause 8.4.3. There are two possibilities of discovering MPE capabilities:

1) EA discovers MPE capabilities through FLUS Control Sink (F1).

2) EA discovers MPE's location through FLUS Control Sink (F1) and discovers the MPE capabilities through N3.

The call flows for both cases are shown below.

8.4.6.2.1. Through F1

Figure 8.4.6.2.1-1 demonstrates the call flow when MPE capabilities are discovered through F1.

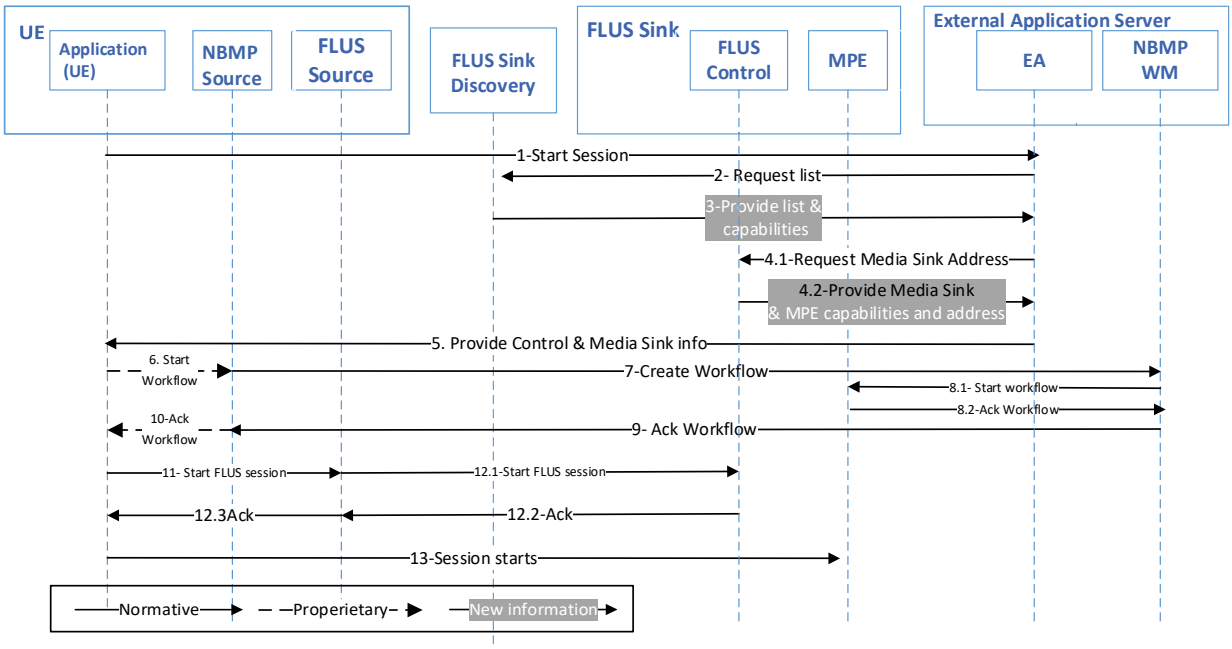


Figure 8.4.6.2.1-1: Call flow for NBMP Client in the UA, NBMP Workflow Manager in the Application Server, and MPE in Sink through F1

The steps of establishing a FLUS-NBMP session are as the following:

1) UE Application (UA) makes a request through F8 to Application (EA) to start a live session.
2) EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server.
3) Sink Discovery Server responds to EA's request.
4) EA picks a FLUS Sink that can run the workflow in its MPE and find its MPE address and MPE APIs in its capabilities.
5) EA responds to UA with FLUS Control Sink and FLUS Media Sink information.
6) UA requests NBMP Client to start an NBMP Workflow with FLUS Media Sink Address.
7) NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.
8) NBMP Workflow Manager instantiates the workflow in the assigned MPE.
9) NBMP Workflow responds to NBMP Client with updated WDD.
10) NBMP Client acknowledges workflow instantiation to EA.
11) UA requests FLUS Control Source to establish the FLUS session.
12) FLUS Control Source establishes the FLUS session and acknowledges UA
13) The session starts.

8.4.6.2.2. Through N3

Figure 8.4.6.2.2-1 demonstrates the call flow when the MPE is discovered through F1 and its capabilities are accessed through N3.

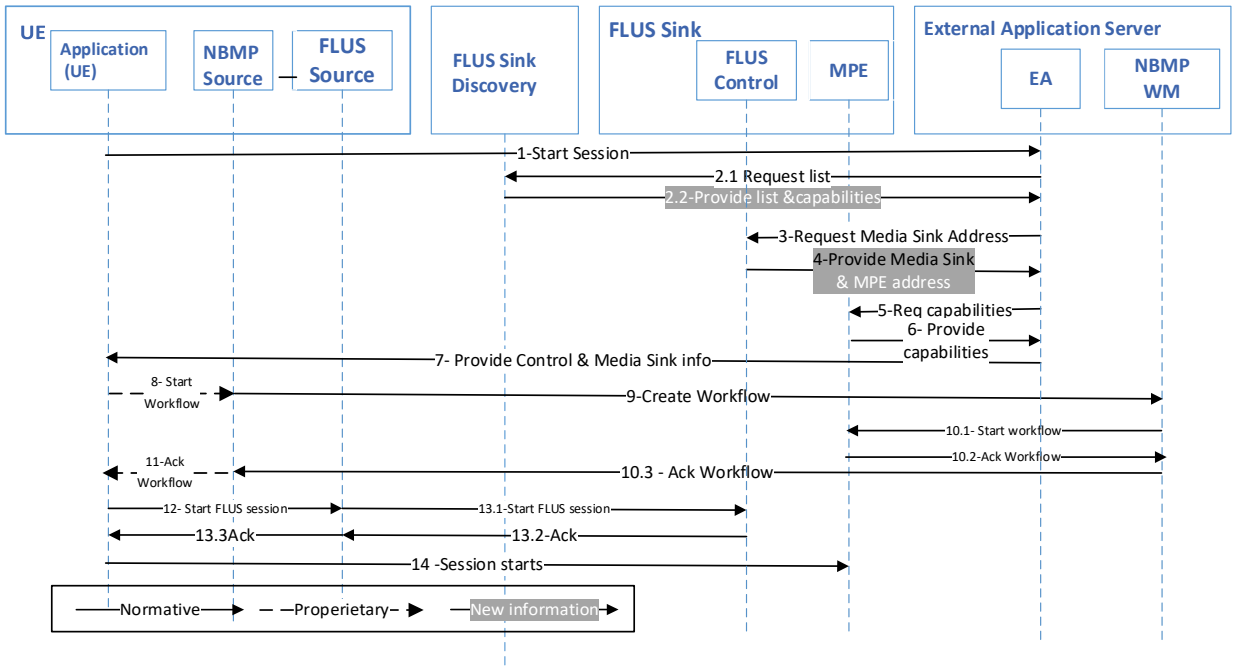


Figure 8.4.6.2.2-1: Call flow for NBMP Client in the UA, NBMP Workflow Manager in the Application Server and MPE in Sink through N3

The steps of establishing a FLUS-NBMP session are as the following:

1) UE Application (UA) makes a request through F8 to Application (EA) to start a live session.
2) EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server and receives it.
3) EA picks a FLUS Sink that can run the workflow in its MPE and request its capabilities.
4) FLUS Sink provides its capabilities to EA including the address of MPE.
5) EA requests the MPE capabilities.
6) MPE provides its capabilities to EA.
7) EA responds to UA with FLUS Control Sink and FLUS Media Sink information.
8) UA requests NBMP Client to start an NBMP Workflow with FLUS Media Sink Address.
9) NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.
10) NBMP Workflow Manager instantiates the workflow in the assigned MPE and acknowledges NBMP Client.
11) NBMP Client acknowledges workflow instantiation to EA.
12) UA requests FLUS Control Source to establish the FLUS session.
13) FLUS Control Source establishes the FLUS session and acknowledges UA
14) The session starts.

8.4.6.3. Interfaces

Table 8.4.6.3-1 shows the required standard interfaces in this scenario.

Table 8.4.6.3-1: Required Standard APIs for NBMP in Application Server, MPE in Sink

Standard	FLUS	F-C, F-U, F1
	NBMP	N4, N3[1]
[1] May be a closed API implemented by the Application Provider-operator agreement.		

NOTE: The internal APIs inside green boxes are out of the scope of this document.

8.4.6.4. Gap analysis

This subclause provides a gap analysis for the above deployment scenario.

8.4.6.4.1. Mapping call flow to the standard APIs

The call flow presented in clause 8.4.6.2.1 is mapped to the FLUS and NBMP APIs in Table 8.4.6.4.1-1:

Table 8.4.6.4.1-1 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)
2. EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server.	Supported by FLUS discovery API.
3. Sink Discovery Server responds to EA's request.	Supported by FLUS discovery API.
4. EA picks a FLUS Sink that can run the workflow in its MPE and find its MPE address and MPE APIs in its capabilities.	Partially supported by FLUS. The MPE URL address is not currently provided in FLUS.
5. EA responds to UA with FLUS Control Sink and FLUS Media Sink information.	Out of scope (optional and application dependent.)
6. UA requests NBMP Client to start an NBMP Workflow with FLUS Media Sink Address.	Out of scope (optional and application dependent.)
7. NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.	For the level of support, please refer to clause 8.4.5.
8. NBMP Workflow Manager instantiates the workflow in the assigned MPE.	Supported by NBMP/exact implementation is MNO specific.
9. NBMP Workflow responds to NBMP Client with updated WDD.	Supported by NBMP.
10. NBMP Client acknowledges workflow instantiation to EA.	Out of scope (Internal to application).
11. UA requests FLUS Control Source to establish the FLUS session.	Out of scope (Internal to application).
12. FLUS Control Source establishes the FLUS session and acknowledges UA	Supported by FLUS.
13. The session starts.	Out of scope (Internal to application).

The call flow presented in clause 8.4.6.2.2 is mapped to the FLUS and NBMP APIs in Table 8.4.6.4.1-2:

Table 8.4.6.4.1-2 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP

1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)
2. EA requests the list of FLUS Sinks and their capabilities from the Sink Discovery Server and receives it.	Supported by FLUS discovery API.
3. EA picks a FLUS Sink that can run the workflow in its MPE and request its capabilities.	Supported by FLUS.
4. FLUS Sink provides its capabilities to EA including the address of MPE.	Partially supported by FLUS. The MPE URL address is not currently provided in FLUS.
5. EA requests the MPE capabilities.	Supported by NBMP.
6. MPE provides its capabilities to EA.	Supported by NBMP.
7. EA responds to UA with Control Sink and Media Sink information.	Out of scope (optional and application dependent.)
8. UA requests NBMP Client to start an NBMP Workflow with FLUS Media Sink Address.	Out of scope (optional and application dependent.)
9. NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow, with the assigned MPE.	For the level of support, please refer to clause 8.4.4.
10. NBMP Workflow Manager instantiates the workflow in the assigned MPE and acknowledges NBMP Client.	Supported by NBMP/exact implementation is MNO specific.
11. NBMP Client acknowledges workflow instantiation to EA.	Out of scope (optional and application dependent.)
12. UA requests FLUS Control Source to establish the FLUS session.	Out of scope (optional and application dependent.)
13. FLUS Control Source establishes the FLUS session and acknowledges UA	Supported in FLUS.
14. The session starts.	Supported in FLUS and NBMP.

8.4.6.4.2. TS 26.238 potential extensions

Tables 8.4.6.4.1-1 and 8.4.6.4.1-2 indicate that the extensions proposed in clauses 8.4.3.4.2 and 8.4.5.4.2 are required to support this deployment scenario.

8.4.7. NBMP Client in the UA, NBMP Workflow Manager, and MPE in the Application Server (NBMPSourceUA-NBMPWMSINK)

8.4.7.1. Architecture

In this scenario as shown in Figure 8.4.7.1-1, NBMP Client is in the UA, NBMP WM and MPE are located in the Application Server.

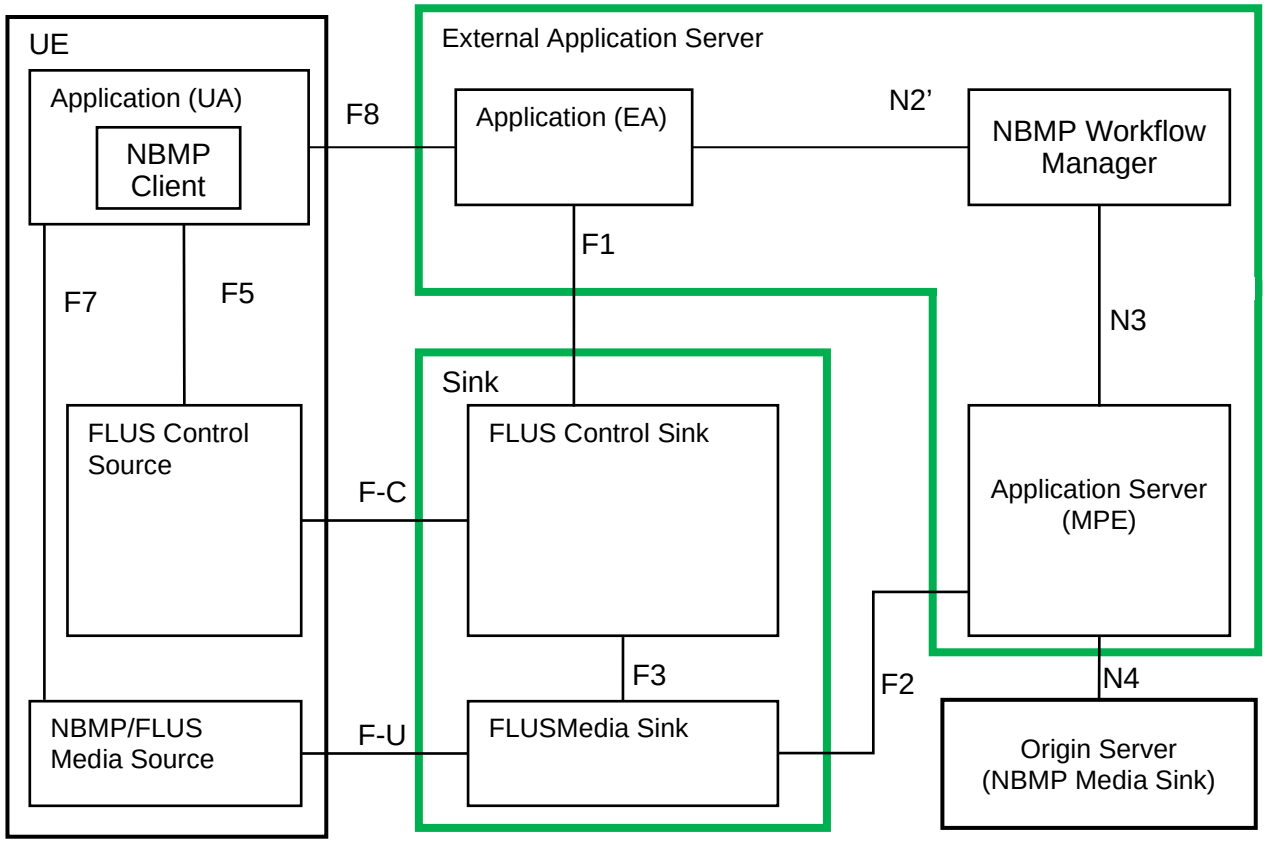


Figure 8.4.7.1-1: NBMP Client in the UA, NBMP Workflow Manager, and MPE in the Application Server

8.4.7.2. Call flow

This scenario is similar to the case when all of NBMP entities are located in the Application Server as described in clause 8.4.2. The call flow is shown in Figure 8.4.7.2-1.

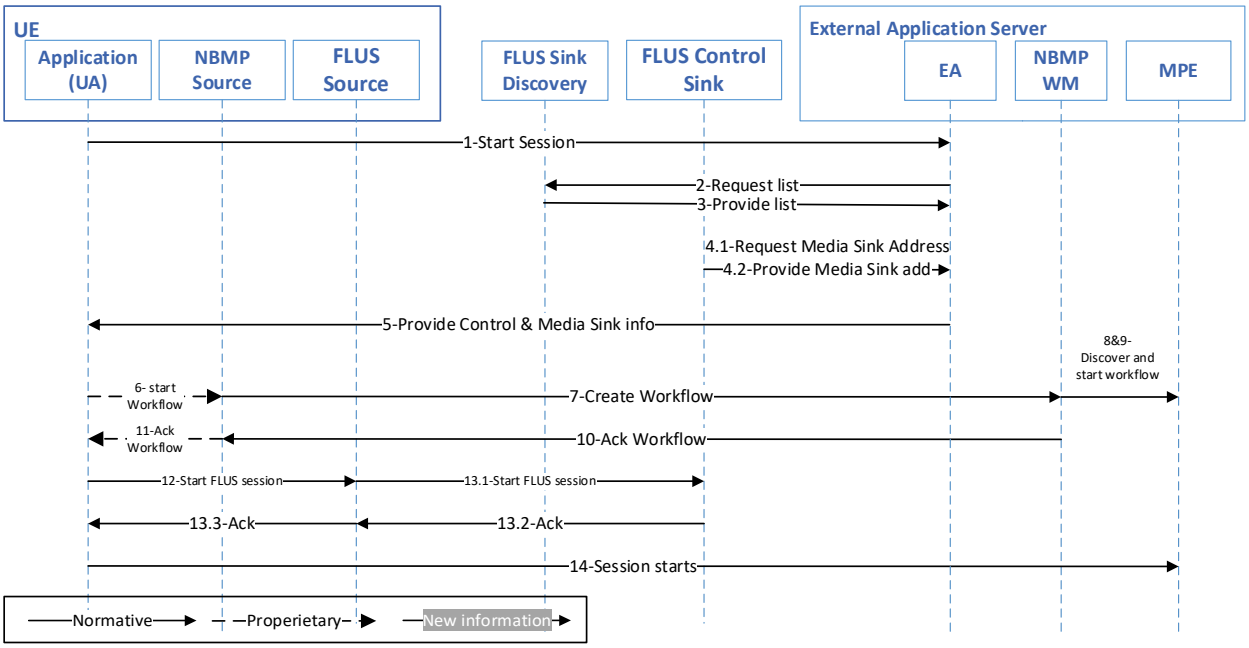


Figure 8.4.7.2-1: Call flow for NBMP Client in the UA, NBMP Workflow Manager, and MPE in the Application Server

The steps of establishing a FLUS-NBMP session are as the following:

1) UE Application (UA) makes a request through F8 to Application (EA) to start a live session.
2) EA requests the list of FLUS Sinks from a Sink Discovery Server.
3) Sink Discovery Server responds to EA's request.
4) EA picks a FLUS Sink and finds its FLUS Media Sink address.
5) EA responds to UA with FLUS Control Sink and FLUS Media Sink information.
6) UA requests NBMP Client to start an NBMP Workflow.
7) NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow.
8) NBMP Workflow Manager discovers various MPEs and finds enough number of MPEs to run the workflow
9) NBMP Workflow Manager instantiates the workflow.
10) NBMP Workflow responds to NBMP Client with updated WDD.
11) NBMP Client acknowledges workflow instantiation to EA.
12) UA requests FLUS Control Source to establish the FLUS session.
13) FLUS Control Source establishes the FLUS session and acknowledges UA.
14) The session starts.

8.4.7.3. Interfaces

Table 8.4.7.3-1 shows the required standard interfaces in this scenario:

Table 8.4.7.3-1: Required Standard APIs for NBMP in Application Server, MPE in Sink

Standard	FLUS	F-C, F-U, F1
	NBMP	N4, F2[1]
[1] The FLUS specification currently does not define setting up an output for FLUS Media Sink. To support this scenario, the FLUS specification needs to be extended to either support setting up an output address for FLUS Media Sink or provide an address for FLUS Media Sink's output for data retrieval.		

NOTE: The internal APIs inside green boxes are out of the scope of this document.

8.4.7.4. Gap analysis

This subclause provides a gap analysis for the above deployment scenario.

8.4.7.4.1. Mapping call flow to the standard APIs

The call flow presented in clause 8.4.7.2 is mapped to the FLUS and NBMP APIs in the following table:

Table 8.4.7.4.1-1 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. UE Application (UA) makes a request through F8 to Application (EA) to start a live session.	Out of scope (optional and application dependent.)
2. EA requests the list of FLUS Sinks from a Sink Discovery Server.	Supported by FLUS discovery API.
3. Sink Discovery Server responds to EA's request.	Supported by FLUS.
4. EA picks a FLUS Sink and finds its FLUS Media Sink address.	Not currently supported by FLUS.
5. EA responds to UA with FLUS Control Sink and FLUS Media Sink information.	Out of scope (Internal to application).
6. UA requests NBMP Client to start an NBMP Workflow.	Supported by NBMP or External Application Provider specific.
7. NBMP Client builds the WDD and requests NBMP Workflow Manager to instantiate the Workflow.	For the level of support, please refer to 8.4.6.4.
8. NBMP Workflow Manager discovers various MPEs and finds enough number of MPEs to run the workflow	Supported by NBMP or External Application Provider specific.
9. NBMP Workflow Manager instantiates the workflow.	Supported by NBMP or External Application Provider specific.
10. NBMP Workflow responds to NBMP Client with updated WDD.	For the level of support, please refer to 8.4.6.4.
11. NBMP Client acknowledges workflow instantiation to EA.	Out of scope (Internal to application).
12. UA requests FLUS Control Source to establish the FLUS session.	Out of scope (Internal to application).
13. FLUS Control Source establishes the FLUS session and acknowledges UA.	Partially support by FLUS as indicated by 8.4.7.3
14. The session starts.	Supported in FLUS and NBMP

8.4.7.4.2. TS 26.238 potential extensions

As shown in table 8.4.7.4.1-1, the required extensions are addressed in clauses 8.4.2.4.2 and 8.4.5.4.2.

8.4.8. NBMP-enabled FLUS Session Establishment using 5GMSu AF (WM-MPE-

Sink-AF)

8.4.8.1. Architecture

This deployment scenario is based on the deployment scenario of clauses 8.4.4 (WM-MPE-Sink) or 8.4.5 (NBMPSource-FLUSSource). However, there exists one or more 5GMS AF that are utilized for provisioning and helping to set up the service. All other deployment scenarios (as described in clauses 8.4.2, 8.4.3, 8.4.6, and 8.4.7) may be extended similarly by utilizing 5GMS AF.

This scenario is shown in Figure 8.4.8.1-1. In this figure, interfaces M1 and M2 indicate the corresponding 5GMSu AF's M1 and M2 interfaces that are defined by TS 26.501, respectively.

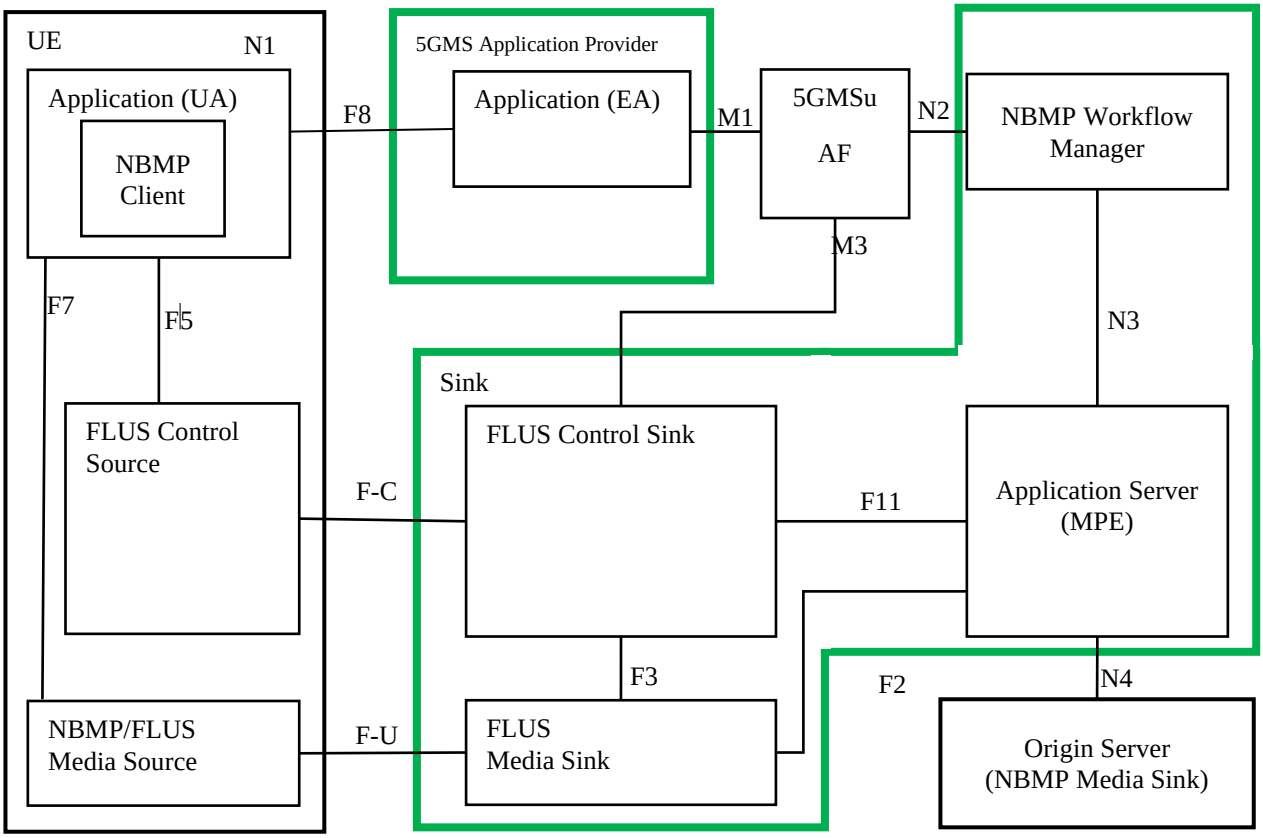


Figure 8.4.8.1-1: NBMP-enabled FLUS Session Establishment using 5GMSu AF

8.4.8.2. Call flow

The FLUS session establishment with NBMP processing is shown in Figure 8.4.8.2-1.

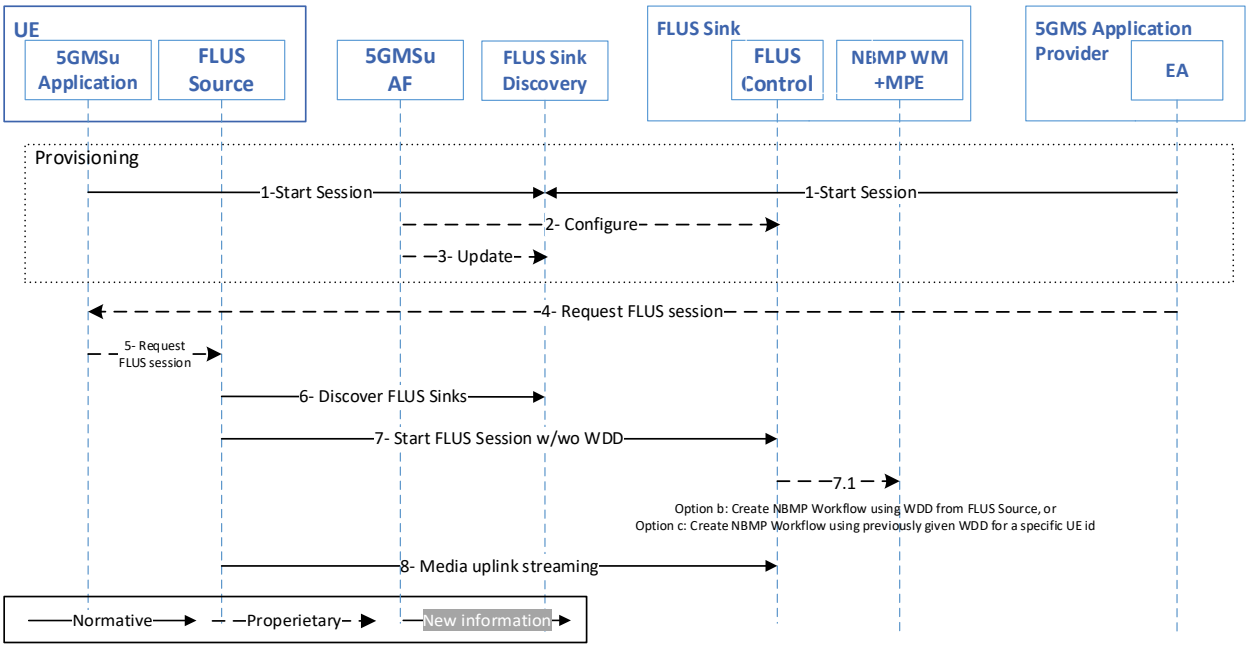


Figure 8.4.8.2-1: NBMP-enabled FLUS Session Establishment using 5GMSu AF

The steps are as follows:

1) The application/application provider establishes a Provisioning session with the AF, in which it configures QoS and Processing templates.
2) The AF configures the FLUS Control Sink(s).
    a. AF may request a FLUS Control Sink to create the NBMP Workflow by providing the WDD to the FLUS Control Sink (option a).
3) The AF updates the FLUS Sink Discovery Server with configured FLUS Sink(s) information.
   NOTE: The Provisioning procedures associated with steps 1-3 are typically performed in advance and not in real time of the FLUS session establishment.
4) The Application Provider signals the Application on UE to start the FLUS Session.
   NOTE: Alternatively, the Application on UE may start the FLUS Session on its own.
5) The Application requests FLUS Control Source to start a FLUS Session.
6) The FLUS Control Source discovers a FLUS Sink that can perform NBMP workflows.
   NOTE: Alternatively, a FLUS Control Source may discover the FLUS Sink from 5GMSu AF through the Media Session Handler of the 5GMSu Client.
7) The FLUS Control Source establishes a connection with the selected FLUS Control Sink.
    a. If NBMP Workflow has been already instantiated by an AF (in 2.a), it only established the FLUS session (i.e., no WDD is sent to the FLUS Sink),
    b. If NBMP Workflow instantiation is requested by Application, then FLUS Control Sink creates the NBMP Workflow according to the WDD received from FLUS Control Source as defined in clause 8.4.4, or
    c. If the FLUS Control Sink already has the UE identifier (such as the GPSI (Generic Public Subscription Identifier)), FLUS Control Source only requests establishing the FLUS session. The FLUS Sink, upon receiving the request, instantiates the Workflow with the previously given WDD.
8) The uplink media streaming starts.
   NOTE: In this deployment architecture, the FLUS Sink is assumed to also include the MPE, and the streaming media output upon NBMP processing by the MPE is provided to the "Distribution" function (see TS 26.238) that is identified by Application Provider/Application.

NOTE: In the above call flow, only one 5GMSu AF is shown for simplicity. In actual deployment, multiple AF instances may be used for different steps of the call flow, e.g. one instance for provisioning, and another instance may be used for setting up other steps.

8.4.8.3. Interfaces

Table 8.4.8.3-1 shows the required standard interfaces in this scenario:

Table 8.4.7.3-1: Required Standard APIs for FLUS-NBMP deployment using 5GMS AF

Standard	FLUS	F-C, F-U
	NBMP	N4, N2[1]
	5GMS	M1, M3
[1] M3 may be used instead of N2.		

NOTE: The internal APIs inside green boxes are out of the scope of this document.

8.4.8.4. Gap analysis

This subclause provides a gap analysis for the above deployment scenario.

8.4.8.4.1. Mapping call flow to the standard APIs

The call flow presented in subclause 8.4.8.2 is mapped to the FLUS and NBMP APIs in the following table:

Table 8.4.8.4.1-1 Mapping call flow to FLUS and NBMP APIs

Call flow step	Support in FLUS or NBMP
1. The application/application provider establishes a Provisioning session with the AF, in which it configures QoS and Processing templates.	Supported by TS 26.512
2. The AF configures the FLUS Control Sink(s).	Out of scope (Internal to MNO)
a. AF may request a FLUS Control Sink to create the NBMP Workflow by providing the WDD to the FLUS Control Sink (option a).	Supported by NBMP or internal MNO's API
3. The AF updates the FLUS Sink Discovery Server with configured FLUS Sink(s) information.	Out of scope (Internal to MNO)
4. The Application Provider signals the Application on UE to start the FLUS Session.	Out of scope (Internal to application).
5. The Application requests FLUS Control Source to start a FLUS Session.	Out of scope (Internal to application).
6. The FLUS Control Source discovers a FLUS Sink that can perform NBMP workflows.	Supported by FLUS.
7. The FLUS Control Source establishes a connection with the selected FLUS Control Sink.	Supported By FLUS.

a. If NBMP Workflow has been already instantiated by an AF (in 2.a), it only established the FLUS session (i.e., no WDD is sent to the FLUS Sink),	Supported by FLUS.
b. If NBMP Workflow instantiation is requested by Application, then FLUS Control Sink creates the NBMP Workflow according to the WDD received from FLUS Control Source as defined in clause 8.4.4, or	For the level of support, please refer to 8.4.4.4.
c. If the FLUS Control Sink already has the UE identifier (such as the GPSI (Generic Public Subscription Identifier)), FLUS Control Source only requests establishing the FLUS session. The FLUS Sink, upon receiving the request, instantiates the Workflow with the previously given WDD.	Supported by FLUS.
8. The uplink media streaming starts.	Support by FLUS and NBMP.

8.4.8.4.2. TS 26.238 potential extensions

As shown in Table 8.4.8.4.1-1, the required extensions are addressed in 8.4.5.4.2.

8.4.9. Summary of the deployment scenarios

Table 8.4.9-1 shows a summary of deployment scenarios.

Table 8.4.9-1: Summary of the deployment scenarios

Scenario	Standard	API
NBMP in Application Server	FLUS	F-C, F-U, F1
	NBMP	N4, F2[1]
NBMP in Application Server, MPE in Sink	FLUS	F-C, F-U, F1
	NBMP	N4, N3[2]
NBMP Client in Application Server, NBMP Workflow Manager, and MPE in Sink	FLUS	F-C, F-U, F1
	NBMP	N2, N4
NBMP Client in FLUS Control Source, NBMP Workflow Manager, and MPE in Sink	FLUS	F-C[3], F-U, F1
	NBMP	N4
NBMP Client in the UA, NBMP Workflow Manager in the Application Server, and MPE in Sink	FLUS	F-C, F-U, F1
	NBMP	N4, N3[2]
NBMP Client in the UA, NBMP Workflow Manager, and MPE in the Application Server	FLUS	F-C, F-U, F1
	NBMP	N4, N2

NBMP-enabled FLUS Session Establishment using 5GMSu AF	FLUS	F-C, F-U
	NBMP	N4, N2[4]
	5GMS	M1, M3

[1] Need an extension to FLUS spec to define output mechanism for FLUS Media Sink.
[2] May be a closed API implemented by the Application Provider-operator agreement.
[3] With the support of NBMP Workflow Manager APIs.
[4] M3 may be used instead of N2.

9 Guidelines for QoS usage for FLUS

9.1 Use-Case introduction

For Live Uplink Streaming, e.g. for professional media production vertical, the 3GPP QoS system needs to strive to 66ehavi throughput requirements of the video flows beyond the guaranteed bit rate.

The Professional Media Production vertical (for example) requires fairly high media bitrates in order to achieve a decent video quality in downlink. In professional media production, uncompressed or lightly compressed video is carried often at speeds of several Gigabit per second (cf. SDI bitrates). This is of course often not feasible for mobile video production, in particular when mobility and wide-area coverage are important features (i.e. when deploying a dedicated LTE cells inside of a media production facility, it could make sense to send uncompressed or lightly compressed frames.

For mobile production, the speed of setting up a live feed (i.e. speed and simplification of production) and the freedom of high mobility is likely more important than high video quality at ultra low latency. Compressed video streams can be used at expense of latency (compression efficiency increases when relaxing latency constrains). Still, the video quality should be high.

The assumption, in the following discussion, is a bitrate adaptive FLUS solution, where the FLUS source can adjust the transmission bitrate to the currently measured / estimated link bitrates. This can be achieved by influencing the encoder bitrate or by dropping frames before transmission.

Figure 9.1-1 below illustrates the desired video quality properties (and the resulting bitrates) as an example.

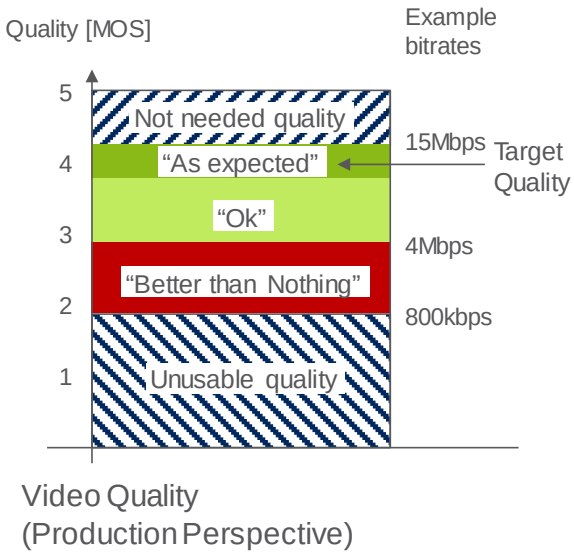


Figure 9.1-1: Quality Principles

The expectation is that the system delivers a certain target quality. Preferably, that target quality is always or as often as possible delivered and the target bitrate should be sustained by the system for a certain time duration. A higher quality

as the target quality is not needed. Depending on the video codec configuration (Codec Profile, codec level and encoder features), the video quality is associated with a bitrate of the compressed stream.

When the system cannot offer the desired target bitrate, then a lower bitrate is acceptable for the video application. The video application layer (e.g. IMS / MTSI, HTTP or others) supports adaptive bitrate adaptation, i.e. it is increasing or decreasing the quality matching whatever link bitrate that is available. In the example above, a resulting video bitrate of ~15Mbps corresponds to the target video quality. The dark green color corresponds to an “as expected quality”. A light green color corresponds to an “ok” quality. The resulting quality is not perfect, but still good to use.

A certain large bitrate range leads to an acceptable quality. The lower end of that bitrate range is the “better than nothing” area, where the video quality contains very obvious quality artifacts. In an example of a media production use-case, the director for the media production may still decide to use the video feed, since the captured pictures are still “better than nothing”. For example, when there was a crash or another event and there is no other video material available.

When the system cannot even offer the lowest quality (here 800kbps), the media producer will terminate the video stream, due to unusable quality. The Video source can stop sending the video stream, since the server is anyhow discarding the content.

The actual quality thresholds depend on the use-cases. The lowest unusable quality threshold is certainly lower for breaking news scenarios than for regular reports. Further, when the camera is mobile, e.g. mounted on a F1 car or a downhill racing skier, the acceptable quality is certainly different than for fixed mounted cameras.

3GPP systems offer different radio access systems. Some radio access systems are capable (depending on the deployment) to provide higher uplink data rates than others. For example, when a device is connected via the new NR radio access network, much higher data rates will be possible than using existing HSPA or GERAN radio access networks.

Figure 9.1-2 below depicts a mobility case, where a mobile uplink streaming client is either getting active in different radio access systems (nomadic mobility) or even moving between access systems with an active uplink streaming session. The different access networks have different bitrate characteristics (of course, deployment release and carrier bandwidth will have similar effects).

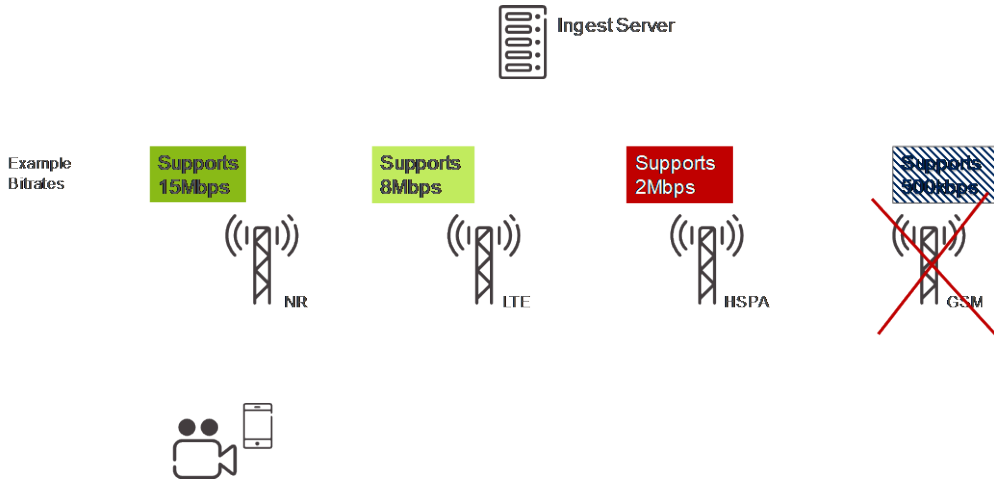


Figure 9.1-2: Mobility and example uplink bitrate expectation

As consequence, there may be handovers within one radio access network (e.g. within NR) or even between radio access networks (e.g. from NR to HSPA).

Due to inter RAT hand-over, the GBR should not be set to a too high bitrate. The UE may handover to a RAT, which does not support such high bitrate and the admission control may reject a QoS bearer. A GBR value should be found, which refers to the bare minimal acceptable bitrate so that each RAT keeps the QoS bearer and the application adapts the bitrate to the admitted parameters.

Beside the mobile media production use-case, there are several other use-cases. The devices may be stationary (e.g. stationary media production or mounted surveillance camera's) and some other may be mobile (e.g. patterns of "breaking news" reporters or vehicle mounted surveillance cameras).

9.2 Discussion of the 3GPP QoS Framework

9.2.1 Introduction

3GPP QoS framework specifies a Guaranteed (Flow) Bitrate (G(F)BR), a Maximum (Flow) Bitrate (M(F)BR), an Allocation and Retention Priority (ARP), and additional QoS Class Indicators (QCI / 5QI). Each QCI defines a priority level (PL), a maximal latency and a maximal packet loss rate for the QoS flow.

9.2.2 Architectures for QoS-based Content Delivery

9.2.2.1 General

In a 5G System (5GS), QoS flows are requested via the PCF (Policy Control Function). In an Evolved Packet System (EPS), QoS bearers are requested via the PCRF (Policy and Charging Rules Function). network entities can directly interact with the PCF / PCRF for QoS. In some cases, for example, depending on business agreements, a network entity such as a 3rd-party operated FLUS Sink acting as an AF (Application Function) might only be permitted to interact with a network mediation function for QoS requests. In 5GS, such mediation function is the NEF (Network Exposure Function) which translates the externally-received parameters to internal network parameters for subsequent interaction with the PCF, on behalf of the network entity, to request/establish QoS-based content delivery. In EPS, the mediation function is the SCEF (Service Capability Exposure Function) which performs similar translation and interaction with the PCRF on behalf of network entities not allowed to directly interact with the PCRF for QoS support.

9.2.2.2 QoS Control in IMS/MTSI-based FLUS

An architecture for an IMS / MTSI-based FLUS system is depicted in Figure 9.2.2.2-1 below. The Session Border Gateway (P-CSCF) forwards the SIP INVITE (call setup message) via potentially other IMS nodes to the FLUS Sink. The SB GW extracts QoS information such as bitrate from the SIP INVITE message (paring the SDP file) and triggers the establishment of a QoS bearer / QoS flow via the PCRF/PCF and subsequent QoS enforcement via the PCEF (Policy and Charging Enforcement Function) or the SMF (Session Management Function). The 5-Tuple(s) for the (uplink) UDP sessions are forwarded as well.

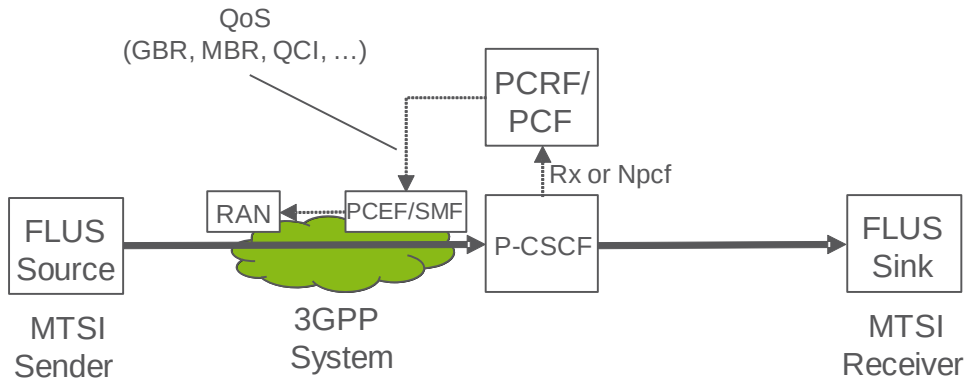


Figure 9.2.2.2-1: IMS / MTSI based architecture (considering QoS terminology)

9.2.2.3 QoS Control in non-IMS/MTSI-based FLUS, Direct Interaction between AF and PCRF/PCF

An example architecture of an HTTP(S)-based FLUS system is depicted in Figure 9.2.2.3-1 below. Here, the FLUS Sink (aka HTTP Server) is able to directly interact with the PCRF/PCF to trigger the establishment of a QoS bearer /

QoS flow, with subsequent QoS enforcement performed by the PCEF/SMF. The FLUS sink needs to wait for the F-U establishment in order to know the 5-Tuple of the session. The FLUS Sink derives the QoS parameters from earlier provisioning steps or from the initialization information of the HTTP FLUS session (i.e. from the existing bitrate ('btrt') box in the codec configuration (e.g. the 'avcC' box for H.264 or a new box for dedicated signaling).

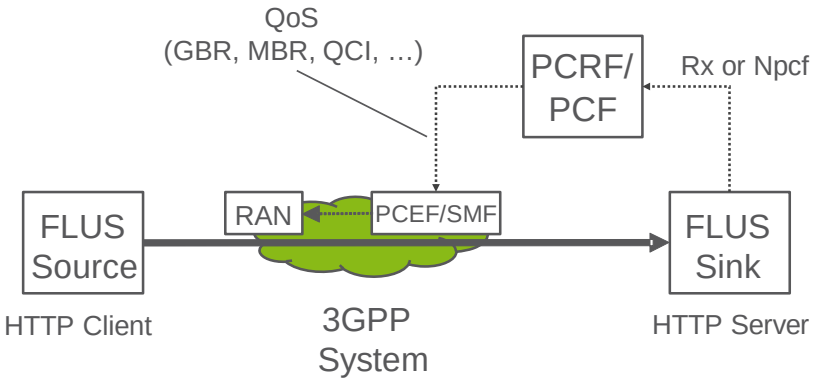


Figure 9.2.2.3-1: HTTP based Architecture (or other OTT protocols), direct interaction between AF and PCRF/PCF

9.2.2.4 QoS Control in non-IMS/MTSI-based FLUS, Indirect Interaction between AF and PCRF/PCF via SCEF/NEF

Another example architecture of an HTTP(S)-based FLUS system is depicted below in Figure 9.2.2.4-1. In this example, the Live Uplink Streaming service, based on FLUS, is assumed to be operated by a 3rd-party application service provider (e.g. Facebook Live). The FLUS Sink acts as an AF and is required to interact with, for QoS support, in the case of EPS, the SCEF (Service Capability Exposure Function) via the T8 reference point, or. In the case of 5GS, for QoS support, the FLUS Sink must interact with the NEF (Network Exposure Function) via the N_{pcf} service-based interface. The SCEF/NEF in turn acts an agent of the AF and proxy of the PCRF/PCF in requesting and receiving approval from the PCRF/PCF for the establishment of a QoS bearer/QoS flow, with subsequent QoS enforcement performed by the PCEF/SMF. The FLUS sink needs to wait for the F-U establishment in order to know the 5-Tuple of the session. The FLUS Sink derives the QoS parameters from earlier provisioning steps or from the initialization information of the HTTP FLUS session (i.e. from the existing bitrate ('btrt') box in the codec configuration (e.g. the 'avcC' box for H.264 or a new box for dedicated signaling).

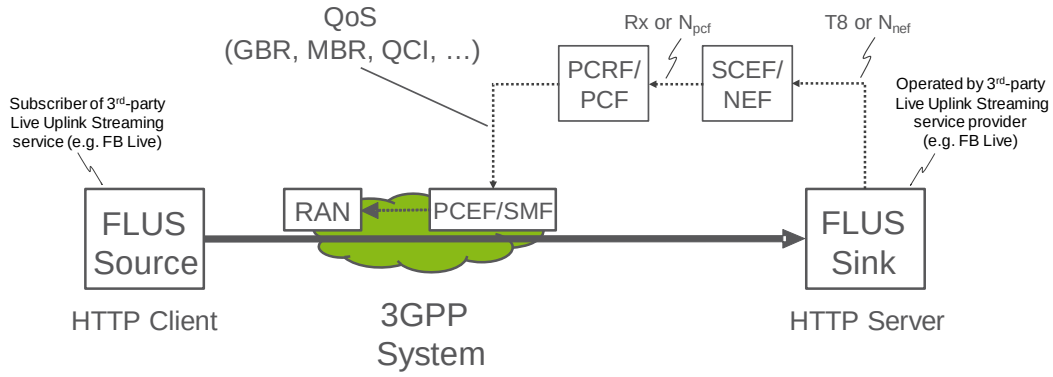


Figure 9.2.2.4-1: HTTP based Architecture (or other OTT protocols), indirect interaction between AF and PCRF/PCF via SCEF/NEF

9.2.3 Relevant 3GPP sections

In the current Rel 15 QoS framework, the Allocation and Retention Priority defines the priority in Admission Control:

5.7.2.2 ARP

The QoS parameter ARP contains information about the priority level, the pre-emption capability and the pre-emption vulnerability. The priority level defines the relative importance of a resource request. This allows deciding whether a new QoS Flow may be accepted or needs to be rejected in case of resource limitations (typically used for admission control of GBR traffic). It may also be used to decide which existing QoS Flow to pre-empt during resource limitations.

The range of the ARP priority level is 1 to 15 with 1 as the highest level of priority. The pre-emption capability information defines whether a service data flow may get resources that were already assigned to another service data flow with a lower priority level. The pre-emption vulnerability information defines whether a service data flow may lose the resources assigned to it in order to admit a service data flow with higher priority level. The pre-emption capability and the pre-emption vulnerability shall be either set to 'yes' or 'no'.

There are two bit rate parameters available to a QoS Flow, GFBR and MFBR:

3GPP TS 23.501 V15.0.0 (2017-12)

5.7.2.5 Flow Bit Rates

For GBR QoS Flows, the 5G QoS profile additionally include the following QoS parameters:

- *Guaranteed Flow Bit Rate (GFBR) – UL and DL;*
- *Maximum Flow Bit Rate (MFBR) – UL and DL.*

The GFBR denotes the bit rate that may be expected to be provided by a GBR QoS Flow. The MFBR limits the bit rate that may be expected to be provided by a GBR QoS Flow (e.g. excess traffic may get discarded by a rate shaping function).

The 3GPP QoS framework leaves the 70ehaviour of the scheduler above the GFBR bit rate value open to implementation:

5.7.3.3 Priority Level

The Priority level indicate a priority in scheduling resources among QoS Flows. The Priority levels shall be used to differentiate between QoS Flows of the same UE, and it shall also be used to differentiate between QoS Flows from different Ues. Once all QoS requirements are fulfilled for the GBR QoS Flows, spare resources can be used for any remaining traffic in an implementation specific manner. The lowest Priority level value corresponds to the highest Priority.

The priority level may be signalled with standardized 5Qis, and if it is received, it overwrites the default value specified in QoS characteristics Table 5.7.4.1. and similarly in 3GPP TS 23.401 V15.2.0 (2017-12):

4.7.3 Bearer level QoS parameters

[…]

Each GBR bearer is additionally associated with the following bearer level QoS parameters:

- *Guaranteed Bit Rate (GBR);*
- *Maximum Bit Rate (MBR).*

The GBR denotes the bit rate that can be expected to be provided by a GBR bearer. The MBR limits the bit rate that can be expected to be provided by a GBR bearer (e.g. excess traffic may get discarded by a rate shaping function). See clause 4.7.4 for further details on GBR and MBR.

4.7.4 Support for Application / Service Layer Rate Adaptation

[…]

The MBR of a particular GBR bearer may be set larger than the GBR.

Note, it would be possible to update the GBR value of a QoS bearer. However, the system does not trigger a renegotiation procedure before dropping a QoS bearer.

9.2.4 Usage of 3GPP QoS parameters

In the following text, the focus is on the GBR/GFBR, the MBR/MFBR and the priority level, since the aim is to get a high sustainable bitrate. Latency configuration of the QCI / 5QI is discussed in sub-clause 9.2.6.

The priority level which is associated to the QCI, is used to differentiate between traffic within a UE and across different Ues up to the GBR (GFBR in 5GC) value ("Once all QoS requirements are fulfilled for the GBR QoS Flows, spare resources can be used for any remaining traffic in an implementation specific manner." [5]) and it does not define a 71ehaviour for a scheduling priority to achieve a "target quality bitrate" larger than GFBR, but less than MFBR, rather only focus on a general resource distribution not related to the useful target bitrate. Moreover, in 4G, the PL parameter is only valid for flows below GBR, and the 71ehaviour of GBR bearers with bitrate above GBR is undefined. Therefore, in many 4G implementations, the GBR bearers will be treated as best effort, or worse, when the bitrate is larger than GBR.

Figure 9.2.4-1: Today's prioritization: traffic gets priorities up to the GBR and is treated as best effort above GBR

The service as introduced in the previous section should typically operate far beyond GFBR/GBR and likely close to MFBR/MBR. If the GFBR/GBR of 3GPP flow/bearer aimed to carry the video traffic is set to the barely acceptable quality level, the scheduling priority will only prioritize the data up to the GFBR/GBR and not really be beneficial to provide bitrates close to the expected service quality. In this case, as the 71ehaviour for traffic between GFBR/GBR and MFBR/MBR is equal to best-effort MBB, then it is probably often better to skip QoS and instead use a non-GBR flow/bearer with high PL (which is likely also cheaper) for the video traffic.

If, on the other hand the GFBR/GBR value of the of 3GPP flow/bearer aimed to carry the video traffic is set to the target quality level, the scheduling priority would lead to the scheduler to prioritize the video traffic up to the target quality level at the cost of more radio resource consumption and reducing the room for the rate adaptation capabilities of the video traffic. While it is clearly desirable to use the target quality, the needed quality/cost trade-off is less optimal in this case, since the cost to guarantee the target quality at all times can easily become too high.

Further, there is an increased risk, that the system is rejecting / dropping the QoS bearer.

9.2.5 Desired QoS flow 71ehaviour

In the following, we discuss the usage of the 3GPP QoS framework.

The system admission control is going to reject / pre-empt a QoS bearer based on the GBR value. In order to get a QoS bearer accepted, the GBR value should be selected as the lowest acceptable bitrate. With increasing GBR value, also the risk is increasing that the system admission control is rejecting / pre-empty QoS bearers based on the GBR value. Note, handovers to other cells / other access networks may retrigger the admission control process.

The MBR is limiting the bitrate of the QoS bearer. In some implementations, the system is dropping traffic when the service bitrate is above MBR. Thus, due to burstiness of video traffic and when a bitrate adaptation principle is available, the MBR should be much larger than the GBR.

The (video) application layer will to tear-down the delivery of the data, when the bitrate (and the resulting quality) falls below the lower threshold, which is indicated as GBR in the Figure 9.2.5-1 below. The preferred service operation point (called target bitrate, TBR) is much higher than the GBR and likely close to the MBR. The FLUS source may adapt the media bitrate to the current estimated link bitrate.

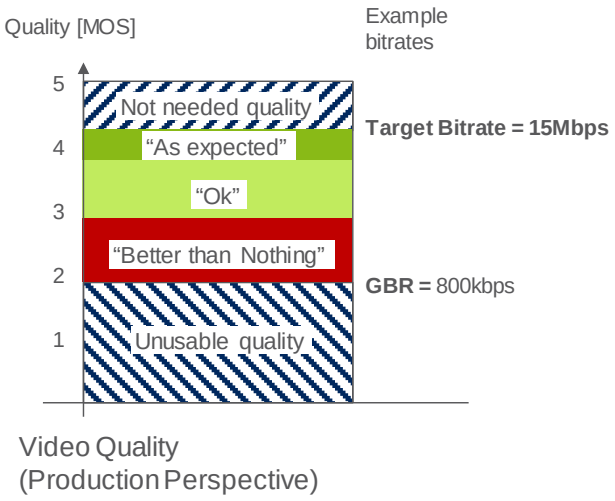


Figure 9.2.5-1: QoS Threshold boundaries

So, a desired behaviour would be when the priority level of a QoS flow does not fall flat to zero once the media bitrate is above GBR/GFBR bitrate. Instead, it would be preferred that the scheduling priority level should decrease gradually with the increasing bitrate. The system should prioritize the QoS flow. The level of prioritization may decrease with increasing media bitrate. As result, the traffic within the QoS bearer would still be treated better than best effort, when the media bitrate is above GBR/GFBR.

Preferable it should be possible to define the priority to get bitrates above GFBR separate from the priority to get the GFBR fulfilled. For the broadcasting media example described here it is prioritized to get high bitrates, but for other services, i.e. public safety, it might be very important to get the GFBR, while higher bitrates have low priority.

9.2.6 Desired QoS latency behavior

9.2.6.1 Typical Latencies for Live Streaming Services

The study in [20] conducted a survey of the latency requirements for live streaming systems, including distribution of the content. The study analyzed the following types of latencies (KPIs):

End-to-End Latency (EEL): The latency for an action that is captured by the camera until its visibility on the remote screen.

Encoding+Distribution Latency (EDL): The latency of the linear playout output (which typically serves as input to distribution encoder(s)) to the screen

Distribution-only Latency (DOL): The latency after the output of the distribution encoder to the screen

CDN latency: The delay caused by the CDN delivery from CDN input to CDN output.

The study then summarized the latency requirements of the various identified use cases as listed in table 9.2.6.1-1.

Table 9.2.6.1-1

Use Case Summary	KPI Impacts
On par with other Distribution Means	*EDL req*, typically as low as **3 seconds up to 10 seconds**. If the delay is larger, the service quality gradually degrades.
ABR competing with Social Feeds originating from the venue	*EEL* typically as low as **5 seconds**
ABR competing with Social Feeds originating from other TV viewers	*EEL* req typically a sum of the above two values, i.e. **8-15 seconds**
ABR competing with Social Feeds originating from other TV viewers	In this case the EEL is typically in between 2A and 2B.
Companion Streams and Screens	*EDL* req, similar to use case 1, but if the delay is larger, than the service typically fails entirely.
Sports Bar	*EDL* req, typically as low as **3 seconds up to 10 seconds**.
Variably Configured Latency across Channels	*EDL* req, typically as low as **3 seconds up to 30 seconds**. However, the latency variations may be used for improved operational performance or reduced cost operations.
DASH in ABR Multicast	The use case does not add any new aspects for the user perception. It more to take into into account conversion from unicast to multicast and reverse. And possible protocol constraints. More a system use case.
DASH as broadcast format	The use case does not add any new aspects for the user perception. It more to take into into account conversion from unicast to multicast and reverse. And possible protocol constraints. More a system use case.
Enterprise Broadcast	*EEL* typically **1/1.5/2s to 5 s**. The delay requirements may be such low as interactivity may be involved.
Hybrid Broadcast	The use case does not add any new aspects for the user perception. It more to take into into account conversion from unicast to multicast and reverse. And possible protocol constraints. More a system use case.
Personal Broadcast	*EEL* typically as low as **5 seconds**. In addition, the use case Single bitrate potentially causing addition buffering limitations.

Channel Bouquet	*EEL* typically configurable between **3 and 30 seconds**. At the same time *start-up and channel change times* are as low as **1-2 seconds**.
Channel Bouquet from one Event	*EEL* typically configurable between **3 and 30 seconds**. At the same time *start-up and channel change times* are as low as **0.5-2 seconds**. Synchronization of the event streams.

The 3-10s of EDL in the first use case represents the typical delay experienced in current TV distribution mechanisms. The above latencies include multiple aspects of the end-to-end latency such as encoding, decoding, rendering, serialization of packets, and the UE to PDN gateway Packet Delay Budget (PDB). The latencies in the table do not directly predict the exact PDB values needed but give an order-of-magnitude reference of the acceptable PDB values (i.e, non-conversational).

The Live services considered in the subsequent subclauses do not include the “camp fire” model where content is recorded much earlier than the distribution, but the distribution is artificially delayed and simultaneously distributed to generate a concurrent viewing experience for the audiences, e.g, talent show performances on TV.

9.2.6.2 Latency of User-Generated Live Uplink Streaming to Social Networks

The study in [21] of a widely used mobile live uplink service revealed a number of relevant observations:

- About 41.5% of user-generated broadcasts to social networks generate no viewers, yet the mobile app uploads the entire content regardless of viewer counts.
- Most of social engagement or viewer interaction with the content occurs after the broadcast has transpired as opposed to during the event. The majority of such interaction occurs one day after the broadcast, and in fact the interactivity volume stays fairly constant for the next eight months after.
- For videos that do generate viewership, on average during the live broadcast, these receive 6.7 likes, 8.4 comments and 0.54 shares. After one day, engagement increases to 29.84 likes, 16.33 comments and 1.33 shares.

Therefore, in some cases of live uplink streaming to social networks, low latency is not required as the content may never be viewed, or is viewed several hours after the upload is completed.

9.2.6.3 Determining New Packet Delay Budgets for 3GPP QCIs and 5QIs

The PDB values specified in QCIs listed in clause 6.1.7.2 of [16] and 5QIs listed in clause 5.7.4 of [6] in Rel-15 are more focused on shorter latencies (5ms, 10ms, 20ms, 50ms, 60ms, 75ms, 100ms, 150ms, 300ms), with many of the lower end values specified for discrete automation, ITS, and remote-control applications. The longer 100ms and 150ms values are used for conversational voice and video, respectively. For the non-conversational (buffered streaming) video, a PDB of 300ms has been defined. All of the existing values are quite limited and overly stringent for the various types of Live Streaming use cases identified in subclause 9.2.6.1.

Therefore, to allow the RAN scheduler to take better advantage of the additional latency tolerance when transporting the higher bit rates on the uplink, new QCIs/5QIs for live uplink streaming of media with the PDB values of 500ms, 1s, 2s, 5s, and 10s were considered.

For some transport protocols, e.g., TCP, the increase in round-trip time (RTT) caused by longer PDB values will decrease throughput. Based on the Mathias et. al. approximation, the steady state TCP throughput is limited as follows:

$$\text{rate} < (\text{MSS}/\text{RTT})*(C/\sqrt{\text{Loss}}) \quad [\ C=1\ ]$$

Setting MSS to 1460 bytes provides the results in Table 9.2.6.3-1.

Table 9.2.6.3-1 TCP Throughput vs. RTT and PLR

RTT (ms)	80	2x150	2x300	2x500	2x1000	2x2000	2x5000	2x10000
Througput (Mbps) @ PLR 1E-06	146	38.93	19.47	11.68	5.84	2.92	1.17	0.584
Througput (Mbps) @ PLR 1E-08	1460	389.33	194.67	116.80	58.40	29.20	11.68	5.84

When Table 9.2.6.3-1 is used with 3GPP QoS PLR and PDB (as ½ RTT) values, the rates roughly estimate worst case throughput as the PDB and PLR values are meant to be maximum limit targets. Even if acceptable for the end-to-end service latency requirements, some of the longer PDB values may not be used for certain transport protocols (e.g., TCP) when needing to stream at higher data rates.

9.2.6.4 Packet Loss Rates

The dependence of TCP throughput on PLR requires investigating PLR values that will work with selected PDB values. RAN level PLR requirements can vary depending on a number of factors, including:

- Use of error recovery transport layer protocols such as TCP or FEC
- Application layer error resiliency (e.g., video IDR and Long-Term Reference frames)
- Codec error resiliency and concealment
- QoE target for the service

In Rel-15, the set of RAN level PLR targets in the QCIs and 5QIs specified in [19] and [6], respectively, are 10^{-2},10^{-3}, 10^{-4},10^{-5}, 10^{-6}. Not all of these PLR values are needed for the uplink streaming use cases. 10^{-4} could be useful for RTP/UDP transport when the service can tolerate this level of packet loss, e.g., audio stream or "low-end video" with slightly better quality than conversational video. On the other hand, as shown in Table 9.2.6.3-1, there is some benefit in specifying a lower PLR value of 10^{-8} in order to support throughput in excess of 40Mbps when using TCP as the transport. Based on this the PLR values of 10^{-4},10^{-6}, 10^{-8} were considered for live uplink streaming.

9.2.6.5 New QCI/5QIs for FLUS

The set of new QCIs and 5QIs listed in Table 9.2.6.4-1 were specified in clause 6.1.7.2 of [19] and in clause 5.7.4 of [6], respectively.

Table 9.2.6.4-1 New QCIs and 5QIs for Live Uplink Streaming

5QI Value	Resource Type	Default Priority Level	Packet Delay Budget	Packet Error Rate	Default Maximum Data Burst Volume (NOTE 2)	Default Averaging Window	Example Services
71		56	150 ms (NOTE 11, NOTE X)	10^{-6}	N/A	2000 ms	"Live" Uplink Streaming (e.g. TS 26.238 [2])
72		56	300 ms (NOTE 11, NOTE X)	10^{-4}	N/A	2000 ms	"Live" Uplink Streaming (e.g. TS 26.238 [2])
73		56	300 ms (NOTE 11, NOTE X)	10^{-8}	N/A	2000 ms	"Live" Uplink Streaming (e.g. TS 26.238 [2])
74		56	500 ms (NOTE 11, NOTE X)	10^{-8}	N/A	2000 ms	"Live" Uplink Streaming (e.g. TS 26.238 [2])
76		56	500 ms (NOTE 11, NOTE X)	10^{-4}	N/A	2000 ms	"Live" Uplink Streaming (e.g. TS 26.238 [2])

9.2.6.6 Media Encoding Latencies

Many of the common use cases for live uplink streaming do not require conversational end-to-end latency and therefore do not require conversational latency for media encoding[1] (e.g, codecs or codec configurations implemented to achieve latencies for conversational services).

It is for further study whether there are commercial/relevant use cases that may require conversational or lower end-to-end latency, and consequently, lower media encoding latency.

10 End-to-End Message Flows

10.1 General

A Live Uplink Streaming ('LUS') service can be subscribed to by one or more users as FLUS Sources to enable uplink streaming delivery to a network-based FLUS Sink, and can be registered by one or more users wishing to discover and register for reception and viewing of the uploaded media content.

The LUS service provider could be a mobile operator, or a 3rd-party application service provider.

10.2 Mobile Network Operator Provided Live Uplink Streaming Service

Figure 10.2-1 illustrates at a high level the end-to-end procedures and interactions between network entities (including the UE-based FLUS Source and media content recipient), for the case of a mobile operator provided LUS service. In

[1] This enables trading off longer encoding latencies achieve better compression efficiency for certain media types.

this example, the downlink delivery method (via unicast/PSS or broadcast/MBMS) is assumed to be determined prior to the uplink streaming event, based on the number and location of interested viewers.

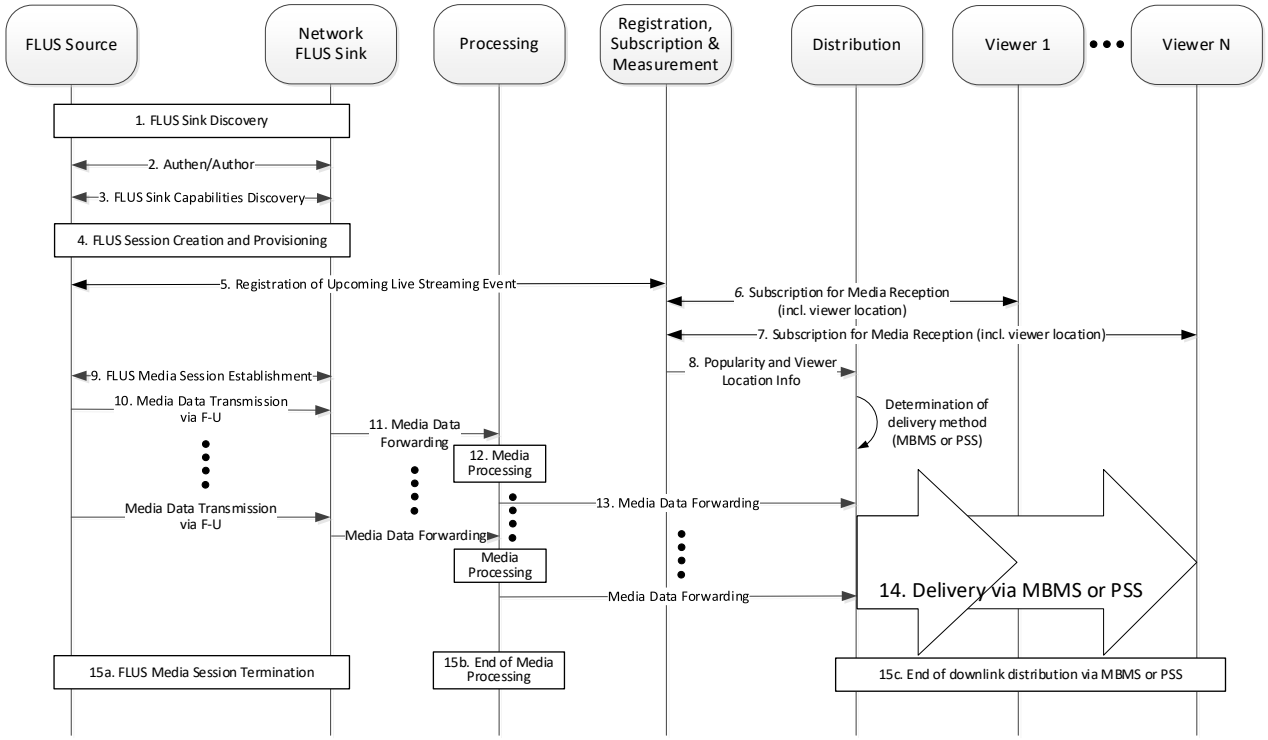


Figure 10.2-1: End-to-end message flow for MNO provided Live Uplink Streaming service

Description of e2e operation of MNO-provided Live Uplink Streaming service:

1) Via F-C, the FLUS sink discovery procedure is performed by the FLUS source to discover available network-based FLUS sinks offered by the FLUS service provider which in this message flow is the mobile operator. Additional information on this procedure can be found in clause 6.1.2 of this document, and normative description of this procedure is provided in clause 7 of TS 26.238 [2].

2) Mutual authentication between FLUS source and FLUS sink, and the FLUS sink checks that the FLUS source is authorized to live stream media content to the network (e.g., from user subscription information).

3) Via F-C, the FLUS sink capabilities discovery procedure is performed by the FLUS source to discover the capabilities of the available (network-based) FLUS sink(s). Additional information on this procedure can be found in clause 6.1.2 of this document, and normative description of this procedure is provided in clause 4.4.4 of TS 26.238 [2].

4) Via F-C, the FLUS source creates a session at the FLUS sink, over which streaming media content can be subsequently sent to the FLUS sink via F-U. Additional information on this procedure can be found in clause 8.2.1 of this document, and normative description of this procedure is provided in clause 7.5 of TS 26.238 [2].

5) The FLUS source interacts with the 'Registration, Subscription & Measurement' function in the network to register the impending live uplink streaming event.

6) and 7). Upon discovery of the schedule of upcoming live streaming events from one or more senders who have registered with the LUS service, viewers 1 to N who have interest in viewing the streaming media content from the FLUS source of concern in this message flow, perform subscription with the Registration, Subscription & Measurement function for reception of that content. It is assumed in this case that the location of each viewer (e.g. cell-ID in which his/her UE is currently located) is also provided to the network in the subscription transaction.

8) Upon receiving and processing the subscriptions from the interested viewers of the upcoming live streaming event associated with the FLUS source, viewership popularity information, along with the location of the

interested viewers are sent from the Registration, Subscription & Measurement function to Distribution function in the network.

Between its reception of the popularity+location information and the time at which the live uplink streaming event occurs, the Distribution function determines whether downlink distribution of the corresponding media content will be done via the broadcast (MBMS) or unicast (PSS) delivery method.

9) Via the FLUS session update procedure over F-C, the FLUS source selects the media session instantiation (e.g., IMS-based uplink transmission, CMAF-based fMP4 transmission, MPU-based fMP4 transmission or RTMP-based transport).

10) At the scheduled “broadcast” time, the FLUS source begins uplink media streaming to the FLUS sink.

11) Forwarding of received streaming content from FLUS sink to network Processing function.

12) Processing of incoming media content such as transcoding, reformatting, media combining, etc. by Processing function in accordance to previous FLUS session configuration (by the FLUS source), network policy, and possibly directives or other information supplied by the Distribution function.

13) Processing function forwards the processed media content to the Distribution functions for downlink delivery to recipient UEs

14) Distribution function performs downlink distribution of the streaming media to viewer devices which have subscribed to receive the LUS service content from the FLUS source, via MBMS or PSS.

15) a-c. At the conclusion of the live streaming event, the FLUS media session is terminated between the FLUS source and FLUS sink, and correspondingly leads to the termination of media processing and downlink distribution activities.

10.3 3rd Party Provided Live Uplink Streaming Service with Downlink Distribution Handlled by Mobile Operator

Figure 10.3-1 illustrates at a high level the end-to-end procedures and interactions between network entities (including the UE-based FLUS Source and media content recipient), for the case of a 3rd-party provided LUS service, for which the 3rd-party provider hands off the uploaded content to a mobile operator for downloink distribution over 3GPP network technology (PSS or MBMS as shown). In this example, similar to Figure 10.2-1, the downlink delivery method (via unicast/PSS or broadcast/MBMS) is assumed to be determined prior to the uplink streaming event, based on information on the number and location of interested viewers provided by the 3rd-party provider to the mobile operator.

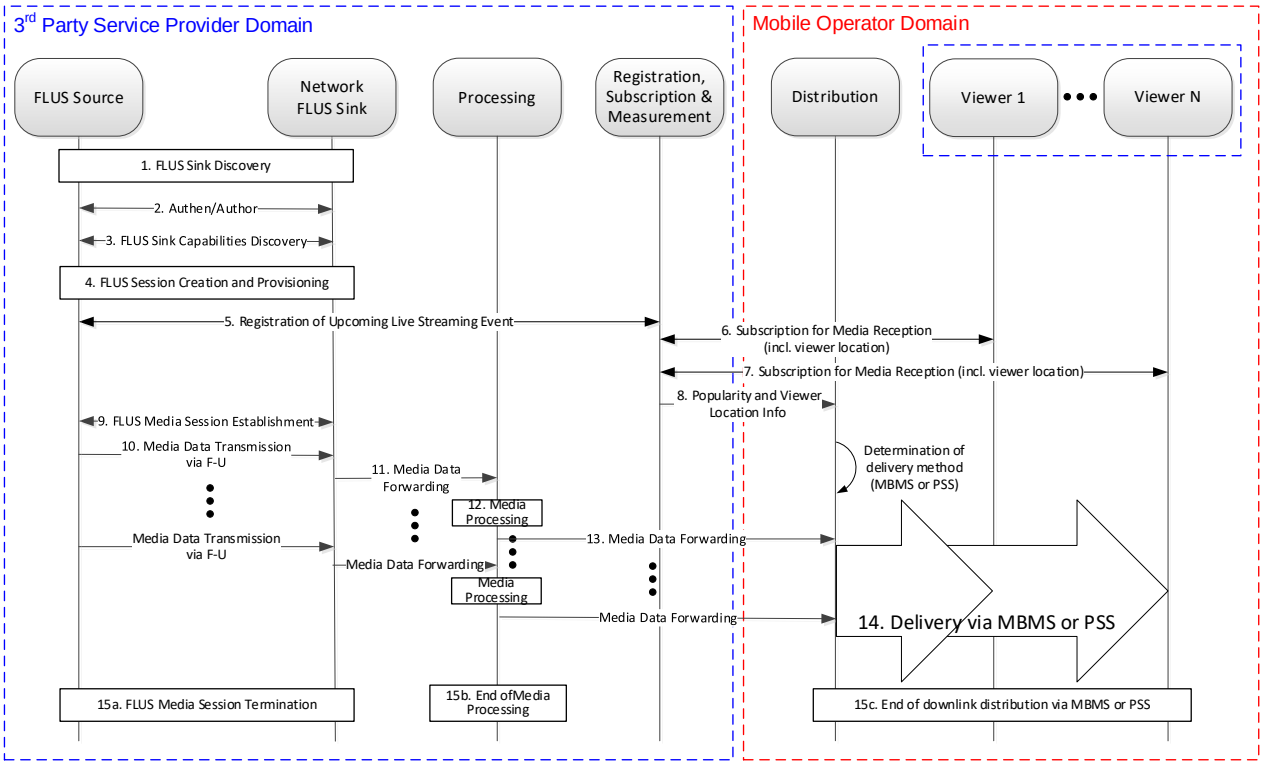


Figure 10.3-1: End-to-end message flow for 3rd Party provided Live Uplink Streaming service with downlink distribution handled by mobile operator

Description of e2e operation of 3rd party-provided Live Uplink Streaming service with downlink distribution handled by MNO:

1-15. The description for each of the numbered transactions between entities in the message flow of Figure 8 is identical to the counterpart numbered step as shown in Figure 10.2-1, with the only differences being: a) in Figure 8, some of the network entities belong to the 3rd-party LUS service provider while the others belong to the mobile operator, while in Figure 7, they all belong to the mobile operator; and b) UEs associated with Viewers 1 to N in Figure 8 are considered to be associated with/belong to both the 3rd-party provider and the mobile operator, while those viewers are strictly associated with/belong to the mobile operator in Figure 7.

11 Guidelines for Uplink Assistance for FLUS

11.1 Uplink Assistance for FLUS based on UNA mechanisms

11.1.1 Introduction

The feature of Network Assistance has been defined so far for 3GPP content downlink services, in particular 3GP-DASH in TS 26.247 [17], including the extension of the 3GP-DASH file format specification in TS 26.244 [18] to adopt the MPEG-SAND messaging construct for its realization. While SAND was developed within the scope of DASH-based content delivery, the general concept of "network-assisted" content delivery is valid for any content container format. Hence this messaging construct is adapted to realise the equivalent functionality for uplink content provision in FLUS.

MPEG-SAND provides a generic message container format and a set of messages. The adaptation and extension of MPEG-SAND to realise downlink Network Assistance for PSS uses largely content container format-independent messages. The only parameter defined within those messages that originates from the principles of DASH is the *segmentDuration* message element. This element is re-purposed to mean any arbitrary regular time period of a content stream, whether in a segmented format or not.

The applicability and usage of Network Assistance for the various FLUS content formats is specified in TS 26.238 [2].

The key functional entity of SAND, namely the DANE (DASH-Aware Network Element), is inappropriately named for the purposes of FLUS Network Assistance. Hence a more appropriate name for the corresponding network element is "Network Assistance Server" (NAssS). This network element is generic, in that in principle it could be used for 3GPP services other than FLUS.

The SAND PER and status messages that are appropriate for usage to realise Uplink Network Assistance are specified in TS 26.238 [2].

As with downlink Network Assistance for PSS, DANE discovery for FLUS Network Assistance will be realised best using the DNS-based discovery method rather than methods that use the *sand:Channel* MPD element, which is in any case a DASH-specific tool.

11.1.2 Network Assistance within the FLUS architecture

Figure 11.1.2-1 shows an example deployment of Uplink Network Assistance (UNA) with respect to the FLUS architecture.

UNA is seen as an accessory to the core FLUS functionality. Furthermore, Network Assistance is seen as a cross-layer feature that is generally applicable to streaming services, both uplink and downlink. The NAssS can be operational in the Operator Network Edge, i.e. in the cell where the FLUS UE is located, or it can be in the Operator Network/Cloud, as is the case with the downlink Network Assistance DANE. The same holds for the FLUS Sink, but independently of the location of the NAssS.

For these reasons the Network Assistance feature for FLUS is incorporated as a separate feature within the FLUS architecture, for which an example deployment configuration is shown in Figure 11.1.2-1. The key aspect is that UNA is a separate optional feature that places a dedicated client function in the FLUS UE and dedicated server function in the operator RAN or core network.

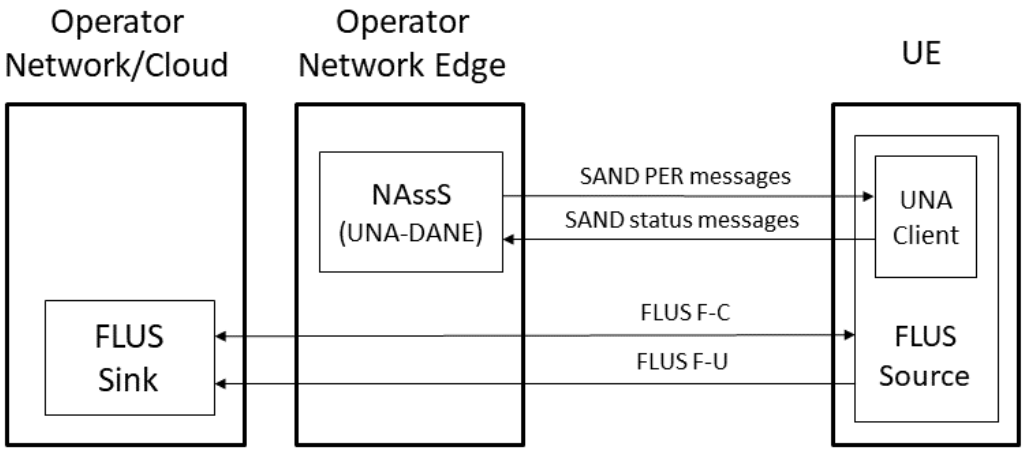


Figure 11.1.2-1: FLUS Network Assistance within the FLUS architecture

11.1.3 Network Assistance for FLUS workflow

Based on the breaking news reporter use case described in clause 6.3, the following workflow is elaborated, in order to set out the basis for the Network Assistance feature for FLUS.

After the camera unit (UE / FLUS Source) has powered up, it proceeds through the sequence of interactions with the network (MNO RAN / NAssS and Content ingestion point / FLUS sink) as depicted in Figure 11.1.3-1, so that it is ready to start streaming the best possible quality version of audio/video coverage under the current local network conditions.

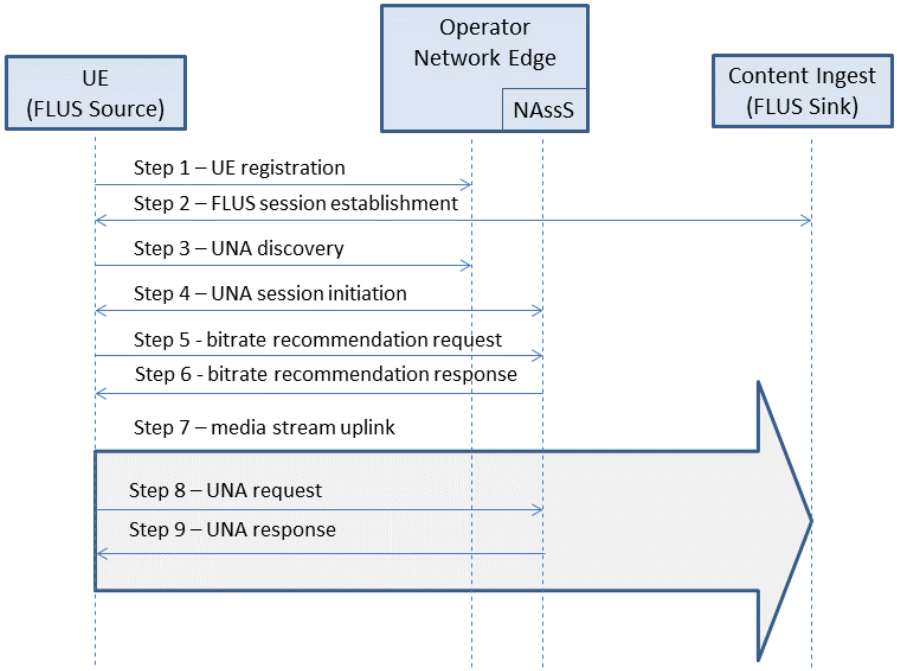


Figure 11.1.3-1: FLUS Network Assistance work flow

1) The UE registers in the Operator Network.
2) The connection is established between the UE (FLUS source) and the TV station (via the FLUS sink) for uplink. This step includes the FLUS session establishment procedure as defined in TS 26.238 [2].
3) The UE uses UNA to establish that Network Assistance is supported at the location.
4) The UE establishes a UNA session with the NAssS that it locates.
5) The UE uses UNA to request a bitrate recommendation for the uplink audio/video stream, to be selected from the versions available in the UE.
6) The NAssS provides the bitrate recommendation.
7) The UE starts/continues the uplink transmission at the recommended bitrate.

Steps 5, 6 and 7 can be repeated at regular intervals during the uplink session.

8) The UE finds that there are transient problems with reaching the FLUS source to supply data packets and the source stream buffer risks getting full. It sends a boost request to the NAssS, in order to attempt to alleviate the content buffer being too full.

The NAssS accepts the boost request and prioritises reception from the UE for the next period of uplink transmission, enabling a higher bitrate than nominally allocated to be transferred for a short period.

11.2 Uplink Assistance for FLUS using RAN Signalling

11.2.1 Introduction

In scenarios where the FLUS QoS does not provide a guaranteed bitrate to the FLUS source, it may benefit the FLUS source to request uplink assistance (i.e., a “boost”) in order to increase its transmission bitrate to the sink when it detects congestion. One solution for providing uplink assistance is to use existing RAN-based signaling (i.e., Access Network Bitrate Recommendation, ANBR) as was specified to support rate adaptation in the MTSI service [16].

The solution has the UE/FLUS source use the already defined ANBR query message to request a boost in the uplink bandwidth provided to the UE. ANBR mapping to the MAC CE message and its usage have already been defined for MTSI. The same mapping would be defined for FLUS where the MAC CE message is mapped to ANBR and ANBR

query in the FLUS specification. In fact, for FLUS, the semantics of ANBR usage are simpler than for MTSI because there is no need to specify procedures for CMR and TMMBR/TMMBN interaction.

11.2.2 Architecture and Call Flow

Illustrated below is how the ANBR-based boost solution can fit into the FLUS architecture and the general call flow.

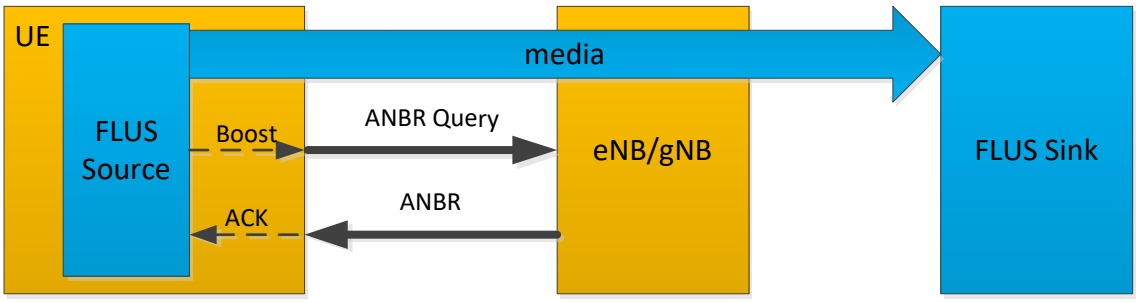


Figure 11.2.2-1: General call flow

The dashed arrows illustrate potential internal messages between the application in the FLUS source and the RAN-level signaling interface in the UE. It is for further study whether to standardize this API in 3GPP, the higher-layer OS, or leave it to the UE vendor implementation.

11.2.3 Semantics

The ANBR query is sent from the UE directly to the eNB/gNB over the air interface using a MAC CE message.

Upon receiving the ANBR Query, the eNB/gNB can send a response to the UE via the ANBR message to indicate what uplink data rate it is granting the UE.

The eNB/gNB can enable a back-off timer that limits how frequently the UE can send consecutive ANBR query messages.

The trigger for the UE to send the ANBR query message can be via an API from the FLUS source, or this could be triggered internally in the UE by its monitoring of the uplink traffic, e.g. checking for backlog in the uplink buffer level.

11.3 Analysis

Following is a comparative analysis of the UNA (Uplink Network Assistance)-based and ANBR-based boost messages.

Boost Message Type	**ANBR (Access Network Bitrate Recommendation)**	**UNA (Uplink Network Assistance)**
Definition	Request and response are already defined in RAN2 – S4 only has to reference	Request and response needs to be defined in SA4
Response Reaction Time measured from the instant the application issues the boost request	UE-eNB/gNB RTT + application-to-UE API delay, as query and response are sent between application and UE via device API, and between UE and eNB/gNB as MAC CE message	UE-eNB/gNB RTT + 2-way application to NAssS HTTP/TCP/IP delay + 2-way NAssS to eNB/gNB interaction delay, in the concatenated path of boost request and response between application and eNB/gNB. The delay between the NAssS and the eNB/gNB depends on the degree of centralized vs. distributed topology of network deployment, i.e., whether the

		NAssS and the eNB/gNB are physically co-located or separated.
Overhead	Minimal overhead as MAC CE message	Additional overhead from HTTP/TCP/IP headers
Back-off solution for boost requests	RAN has already defined back-off timer for requests	Need to define this in SA4
API	Either the UE autonomously detects the need for ANBR-Query (e.g., monitors uplink buffer levels assuming that it has access to that info) or requires application communication to the UE to trigger ANBR-Query. Communication can be performed over a non-standardized or standardized API. If standardized API is desired, can leverage semantics of MBMS API [22].	Requires communication between NAssS and eNB/gNB over a non-standardized or standardized network interface, or could make use of network QoS APIs (e.g, Rx/T8, Nnef/N33) updated to support dynamic boost requests.

12 FLUS Architecture Including UE-based Control Point and FLUS Remote Assist/Control

12.1 FLUS Architecture Considering UE-based Control Point

The current FLUS system architecture, as copied from TS 26.238, is depicted in Figure 12.1 below.

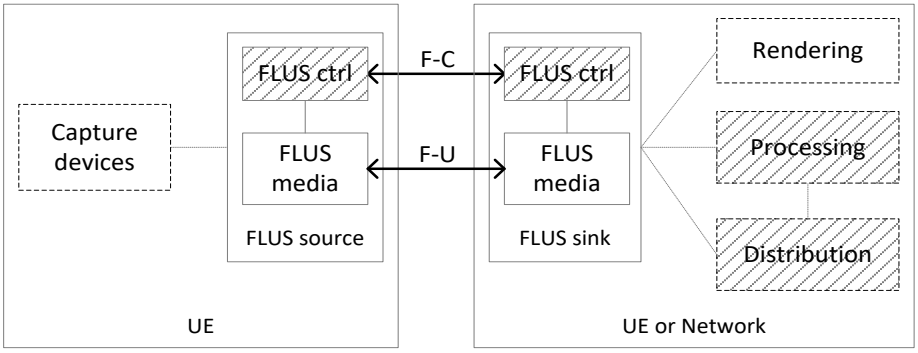


Figure 12.1: Generic FLUS Architecture (identical to Fig. 4.4-1 of TS 26.238 [2])

As indicated in TS 26.238, a single FLUS control-related function and interface, F-C, is used to establish and control the FLUS session, and allows the FLUS source to perform FLUS sink discovery, capability exchange, session establishment and update, selection of a FLUS media instantiation, provisioning of static metadata associated with each media session present in the FLUS session, selection and configuration of network processing and distribution sub-functions, and session termination between the FLUS source and the FLUS sink. To-date, the FLUS architecture does not consider or depict data sources which reside in the network such as core network or RAN entities, or pertain to specific application management systems (e.g. audience measurement) and can provide relevant information to the UCP for guiding or controlling upload behavior by the FLUS source.

An example of a modified FLUS system architecture which leverages a so-called “FLUS Remote Assist/Control” function and associated interface in support of delivering assistance information to the UCP for media upload control, along with the UCP entity itself, is shown in Figure 12.2.

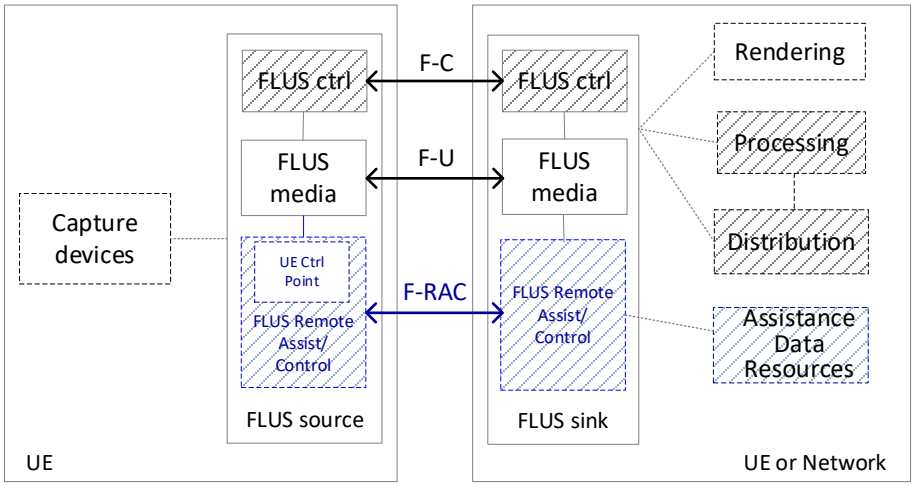


Figure 12.2: Revised FLUS Architecture showing UCP and Data Sources

In the diagram, it is assumed that the data pertaining to network/systems status or other assistance information are collected by the FLUS Remote Assist/Control function residing in the network and delivered to the UCP via the F-RAC interface. Specifically, the network-side data sources, represented by the "Assistance Data Resources" functional entity, feed information to the FLUS Remote Assist/Control entity in the network FLUS sink, which in turn makes that information available via F-RAC to the UCP logical function inside FLUS Remote Assist/Control to affect FLUS media upload delivery behavior over F-U. Note that the Assistance Data Resources box in the diagram is an abstraction for a set of one or more data sources which provide various types of information, as described in clause 12.2.2 below, to the UCP in support of FLUS media control.

12.2 Functionality to be Supported by F-RAC

12.2.1 Remote Command/Control Information

As described in the sub-clauses of clause 8, FLUS remote control functionality, supported by the F-RC interface can support a control entity located externally from the FLUS source to send command such as "start" and "stop" to remotely control the start and end of uplink streaming operation by the FLUS source. Other candidate remote commands might include pan, zoom or tilt functionality of the UE camera.

12.2.2 Remote Assistance Information Collection and Transfer

As indicated in clause 6.6.1.2, the following types of information are relevant for use by the UCP for controlling or guiding the uplink streaming behavior of the FLUS media function in the FLUS source:

- User preference on minimizing unnecessary uplink data traffic;
- Congestion or high load of the network processing subsystem of the FLUS system;
- Downlink distribution method selected by the FLUS system;
- Measured or predicted viewership for the uplink streaming content;
- Measured or predicted interactive engagement with the uplink streaming content;
- Status on access network such as available access network types (Wi-Fi, 3G, 4G, 5G), and specific to 3GPP RAN technology: availability of excess bandwidth, non-GBR capacity;
- Location of the UE in the serving cell coverage area relative to the base station.

The message format, syntax and semantics of remote assistance related messages to be delivered over the F-RAC interface as shown in Figure 12.2 from the networkwill need to be specified. In addition, explicit procedures pertaining to the collection of such assistance information from various data sources, by the network-resident FLUS remote assist/control function, needs to be defined.

Annex A:
Immersive media signalling

A.1 General

This clause defines the parameters for signalling immersive media sessions. A media client may support all or a subset of these parameters, and the receiver of SDP may ignore the parameters it cannot understand or whose usages do not comply with the definitions. These parameters, which are associated with a media-level attribute, mediagmtr, may be used with RTP or other transport protocols. The reference coordinate system for these parameters is defined in [2].

The syntax for the attribute, following ABNF, is as follows:

media-geometry = “mediagmtr:” PT 1*2 (1*WSP (“send” / “recv”) 1*WSP attr-list) 1*WSP “vp-depend=“ vp-depend

PT = 1*DIGIT / “*”

attr-list = (set *(1*WSP set)) / “*”

; WSP and DIGIT are defined in [11]

set = “[“ “az=“ az “;” “el=“ el “]”

NOTE 1: Syntax of az and el depends on the payload type they specify, due to the intrinsic differences between media. If they are used for audio, definitions in clause B.1 are used. If they are used for video, definitions in clause B.2 are used.

NOTE 2: If both azimuth and elevation angles are specified for audio, the numbers of angles are identical.

A.2 Audio

The following parameters are applicable in sessions including channel-based audio.

Az: specifies the azimuth angles of audio channels in degrees, for the send or receive direction. The parameter can have a single angle or a comma-separated list of angles, and each angle is a real number greater than or equal to -180 but less than or equal to 180.

El: specifies the elevation angles of audio channels in degrees, for the send or receive direction. The parameter can have a single angle or a comma-separated list of angles, and each angle is a real number greater than or equal to -90 but less than or equal to 90.

Vp-depend: Permissible values are 0 and 1. If vp-depend is 0, the audio signals are captured in fixed directions. If vp-depend is 1 and information on the direction of viewport at the receiver is available at the sender, the audio signals are captured in directions taking the direction of viewport into account.

NOTE: The audio signals are assumed to approach the origin in the specified directions.

A.3 Video

The following parameters are applicable in sessions including video.

Az: specifies the range of azimuth angle for video in degrees, for the send or receive direction. The parameter can have a hyphen-separated pair of two angles (az1-az2), and each angle is a real number greater than or equal to -180 but less than or equal to 180. Az1 is smaller than az2.

El: specifies the range of elevation angle for video in degrees, for the send or receive direction. The parameter can have a hyphen-separated pair of two angles (el1-el2), and each angle is a real number greater than or equal to -90 but less than or equal to 90. El1 is smaller than el2.

Vp-depend: Permissible values are 0 and 1. If vp-depend is 0, the video signals are captured in fixed directions. If vp-depend is 1 and information on the direction of viewport at the receiver is available at the sender, the video signals are captured in directions taking the direction of viewport into account.

NOTE 1: The video signals are assumed to be projected on the internal surface of a spherical display.

NOTE 2: If neither azimuth nor elevation angle is specified, the video signals are assumed to be projected on a flat display whose resolution is specified by the imageattr attribute.

A.4 Examples of SDP offers and answers

A.4.1 H.264 (AVC), H.265 (HEVC), and EVS

The SDP offer includes H.264/H.265 for video and EVS for audio. In this example, direction of audio channels and range of video viewport are negotiated. Although two video codecs are offered, only H.265 is considered for a spherical display. EVS is offered as a dual-mono configuration with DTX disabled and audio bandwidth maximized to fullband, and also as a conventional configuration for super-wideband telephony. It is assumed that further information related to media handling, e.g., parameters on the projection or packing of video, is signalled using other methods.

Table A.4.1: Example SDP offer

SDP offer
m=audio 49152 RTP/AVP 97 98 b=AS:146 b=RS:0 b=RR:2000 a=rtpmap:97 EVS/16000/2 a=fmtp:97 br-send=64; bw-send=nb-fb; ch-send=2; dtx=0; max-red=220 a=mediagmtr:97 send [az=-40,40;el=45,45] vp-depend=1 a=rtpmap:98 EVS/16000/1 a=fmtp:98 br-send=5.9-24.4; bw-send=nb-swb; max-red=220 a=ptime:20 a=maxptime:240 a=sendonly m=video 49154 RTP/AVP 99 100 b=AS:15000 b=RS:0 b=RR:5000 a=rtpmap:99 H265/90000 a=fmtp:99 profile-id=1; level-id=51 a=imageattr :99 send [x=3840,y=2160] a=mediagmtr:99 send [az=-180-180;el=-90-90] vp-depend=1 a=rtpmap:100 H264/90000 a=fmtp:100 packetization-mode=0; profile-level-id=42e01f a=imageattr :100 send [x=640,y=480] a=sendonly

A maximum of 146 kbps is offered for a dual-mono configuration of EVS at 64 kbps. Two audio channels are offered in two directions that share the same elevation. Although not shown in the offer, an alternative of 42 kbps is also provided for a mono configuration at bit-rates up to 24.4 kbps. These are offered as one-way transmission from the media sender. In addition, a maximum of 15 Mbps is offered for an omnidirectional 4K video encoded with H.265. Level 5.1 of this codec supports resolution and frame rate up to 4K and 60 fps respectively. An alternative is a lower-resolution video encoded with H.264, which is expected to be projected on a flat display.

Table A.4.2: Example SDP answer

SDP answer
m=audio 49152 RTP/AVP 97 b=AS:146 b=RS:0 b=RR:2000 a=rtpmap:97 EVS/16000/2 a=fmtp:97 br-recv=64; bw-recv=nb-fb; ch-recv=2; dtx=0; max-red=220 a=mediagmtr:97 recv [az=-40,40;el=45,45] vp-depend=1 a=ptime:20 a=maxptime:240 a=recvonly m=video 49154 RTP/AVP 99 b=AS:10000 b=RS:0 b=RR:5000 a=rtpmap:99 H265/90000 a=fmtp:99 profile-id=1; level-id=51 a=imageattr :99 recv [x=3840,y=2160] a=mediagmtr:99 recv [az=-120-120;el=-90-90] vp-depend=1 a=recvonly

In the SDP answer, from the offered media configurations, a dual-mono configuration of EVS at 64 kbps and a 4K video encoded with H.265 were selected. In the case of video, the bit-rate is reduced to 10 Mbps as the range of video viewport is reduced by a third, i.e. 360 to 240 degrees, in azimuth. The directions in the attributes and parameters are all reversed. Figure B.3.1 illustrates the geometry of audiovisual media negotiated.

The user is assumed to be located inside the partial sphere. The video is projected on the internal surface of the sphere, and the two arrows represent the direction of audio channels. As both configurations were offered and answered with vp-depend=1, the media sender will take the received information on the video viewport at the media receiver, if available, into account.

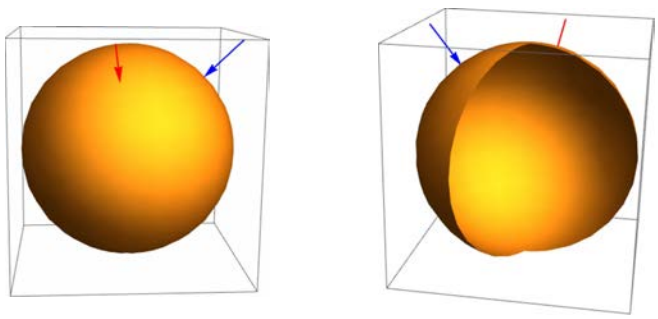


Figure A.4.1: Negotiated media geometry

Annex B:
Change history

Change history							
Date	**Meeting**	**Tdoc**	**CR**	**Rev**	**Cat**	**Subject/Comment**	**New version**
06-2018	SA#80	SP-180267				Presented to TSG SA#80 (for approval)	1.0.0
06-2018	SA#80					Approved at TSG SA#80 plenary meeting	15.0.0
09-2018	SA#81	SP-180640	0001	2	F	Corrections of fMP4 instantiation description	15.1.0
06-2019	SA#84	SP-190341	0002	1	B	E-FLUS Updates	16.0.0
09-2019	SA#85	SP-190651	0003	1	D	Proposed Improvements to Drone-Mounted Camera Architecture Description	16.1.0
09-2019	SA#85	SP-190651	0005	2	B	Fisheye Omnidirectional Video Use Case	16.1.0
09-2019	SA#85	SP-190651	0006	1	F	Modifications to Use Case Descriptions under Clause 6.5	16.1.0
09-2019	SA#85	SP-190651	0007	1	B	New Use Case and Correction for E_FLUS	16.1.0
09-2019	SA#85	SP-190651	0008	1	F	Media Production Use Case	16.1.0
09-2021	SA#93-e	SP-210827	0011	1	B	Support of Network-Based Media Processing	17.0.0
2024-03	-	-	-	-	-	Update to Rel-18 version (MCC)	**18.0.0**

History

Document history		
V18.0.0	May 2024	Publication