**3GPP TSG-SA4 Meeting #131 *S4-250269***

**Geneva, Switzerland, 17 Feb 2025 – 21 Feb 2025 Revision of S4-250080**

|  |
| --- |
| *CR-Form-v12.3* |
| **CHANGE REQUEST** |
|  |
|  | **26.264** | **CR** | **0002** | **rev** | **1** | **Current version:** | **18.1.0** |  |
|  |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME | **X** | Radio Access Network |  | Core Network | **X** |

|  |
| --- |
|  |
| ***Title:***  | Remove MRF from IMS data channel architecture  |
|  |  |
| ***Source to WG:*** | Samsung Electronics, CO., LTD |
| ***Source to TSG:*** | S4 |
|  |  |
| ***Work item code:*** | IBACS, TEI18 |  | ***Date:*** | 2025-02-12 |
|  |  |  |  |  |
| ***Category:*** | **F** |  | ***Release:*** | Rel-18 |
|  | *Use one of the following categories:****F*** *(correction)****A*** *(mirror corresponding to a change in an earlier release)****B*** *(addition of feature),* ***C*** *(functional modification of feature)****D*** *(editorial modification)*Detailed explanations of the above categories canbe found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | *Use one of the following releases:Rel-8 (Release 8)Rel-9 (Release 9)Rel-10 (Release 10)Rel-11 (Release 11)…Rel-17 (Release 17)Rel-18 (Release 18)Rel-19 (Release 19) Rel-20 (Release 20)* |
|  |  |
| ***Reason for change:*** | Latest version of the stage 3 work item on NG\_RTC as approved in CP-240271 at CT#103 has removed MRF as a supporting node for DC, i.e. MRF is not supported as media function in stage 3 Rel-18. CT1 agreed CR (C1-243852) to remove MRF from the relevant TS 24.186 and informed SA2 about the removal to align SA2 specification accordingly.  |
|  |  |
| ***Summary of change:*** | * Remove MRF from IMS data channel architecture.
* Further editorial changes.
 |
|  |  |
| ***Consequences if not approved:*** | Technical specification not aligned with stage 2/3 works from SA2 and CT1 groups |
|  |  |
| ***Clauses affected:*** | 3.1, 3.3, 4.1, 4.2, 4.3, 6.1, 6.3.1.2, 6.4, 7.1, 7.2, A.1.3, A.1.4, A.1.5 |
|  |  |
|  | **Y** | **N** |  |  |
| ***Other specs*** |  | **X** |  Other core specifications  | TS/TR ... CR ...  |
| ***affected:*** |  | **X** |  Test specifications | TS/TR ... CR ...  |
| ***(show related CRs)*** |  | **X** |  O&M Specifications | TS/TR ... CR ...  |
|  |  |
| ***Other comments:*** |  |
|  |  |
| ***This CR's revision history:*** | Rev 1 (S4-250269) : Updates in SA4#131 (Missing list of affected clauses, Update of Figure 4.3.1, Font style of ABNF syntax) |

\* \* \* \* Change #1 \* \* \* \*

# 3 Definitions of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms given in 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 TR 21.905 [1].

**AR data:** Collection of information to be exchanged among participants in a call with AR experience. It includes AR media and AR metadata.

**AR media:** Media (e.g., audio, video, text or image) that will be rendered by the AR-MTSI client as an overlay over the user’s real perception. This includes traditional 2D media (e.g., a 2D audio stream rendered to be perceived by the user to originate from their left side) and 3D media (e.g., spatial audio and volumetric video).

**AR metadata:** Data that provides information on AR media and its rendering. This includes pose, spatial descriptions and scene descriptions.

**AR-MTSI client:** A DCMTSI client supporting AR capabilities as defined by this specification.

**AR MF:** An AR-MTSI client implemented by functionality included in the MF.

**AR-MTSI client in terminal:** An AR-MTSI client that is implemented in a terminal or UE. The term "AR-MTSI client in terminal" is used in this document when entities such as AR MF/MRF is excluded.

**Split rendering**: The procedure in which a UE offloads some of the media processing related to rendering tasks to a media function as considered for network centric AR IMS session procedures in TS 23.228 [4]

## 3.2 Symbols

Void

## 3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 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 TR 21.905 [1].

ADC Application Data Channel

BDC Bootstrap Data Channel

AS Application Server

DC Data Channel

DCSF Data Channel Signalling Function

I‑CSCF Interrogating‑CSCF

IMS IP Multimedia Core Network Subsystem

MF Media Function

P‑CSCF Proxy‑CSCF

S‑CSCF Serving‑CSCF

\* \* \* \* Change #2 \* \* \* \*

# 4 System description

## 4.1 General

Typical conversational AR scenarios as envisioned in this document consist of an immersive AR call that may include the following conversational components:

- Real-time speech/audio that can comprise mono, stereo, and/or spatial audio.

- Real-time 2D video or 360-degree video that can be rendered as rectangular or spherical overlay in the AR experience.

- A real-time volumetric video of the user or an object that can be rendered in AR or MR.

In addition to the above conversational media, non-real-time objects may be exchanged over the data channel as well.

Both two-party and multiparty calls are possible. The AR experience may be unidirectional, i.e., only one party receives AR media and renders it, or it may be bidirectional, i.e., both parties receive and transmit AR media. The term AR-MTSI client includes both:

- an AR-MTSI client in terminal which is an AR device as defined in TS 26.119 [3] e.g., AR glasses, phone, Head Mounted Display (HMD) that has an XR Runtime for rendering an AR experience.

- AR MF that provides support for AR conversational services.

As an AR-MTSI client in terminal is a DCMTSI client in terminal with additional features for AR communication, the following requirements for a MTSI client terminal also apply for an AR-MTSI client in terminal:

- the interworking requirements in clause 12 of TS 26.114 [2],

- the jitter buffer management requirements in clause 8 of TS 26.114 [2],

- the packet loss handling requirements in clause 9 of TS 26.114 [2],

- the media and rate adaptation requirements in clause 10 and 17 of TS 26.114 [2], and

- the network preference management object in clause 15 of TS 26.114 [2],

NOTE: If an AR-MTSI client in terminal supports functionalities for MSMTSI client in terminal as specified in Annex S of TS 26.114 [2], the media and rate adaptation requirements in Annex S.8 of TS 26.114 [2] also apply for an AR-MTSI client in terminal.

## 4.2 Terminal architecture

The detailed XR client architecture is not in the scope of this specification. The XR baseline client architecture can be found in TS 26.119 [3]. The pre/post-processor component in terminal provides AR capabilities for processing output of peripherals and the input/output of encoders/decoders, which may include:

- XR runtime

- Scene manager

- Presentation engine

- XR source management

The AR-MTSI client has XR Runtime capabilities for rendering AR experience, e.g., spatial localization and mapping, etc., and can support local AR rendering and network-assisted split rendering based on client’s capabilities. A UE may support multiple microphones, cameras or sensors.

An AR-MTSI client supports the protocol stack of a basic MTSI client as described in clause 4.2 of TS 26.114 [2]. For the specific AR communication instance, AR-MTSI client can select different IMS media channel to deliver AR data to IMS network or peer UE. In general, AR media components with real-time characteristics are transported via RTP session and AR metadata is transported via data channel or RTP session with AR media.



Figure 4.2.1: Functional components of an AR-MTSI client in terminal

## 4.3 End-to-End Reference Architecture

The end-to-end architecture to support AR communication over IMS can be found in TS 23.228 Annex AC [4]. The following Figure 4.3.1 is a simplified version showing the media functions within the scope of this specification.



Figure 4.3.1: Generalized IMS DC Architecture to support AR communication

NOTE 1: General control-related elements over Gm interface, such as SIP signalling (TS 24.229 [5]), fall outside the scope of this specification, albeit parts of the session setup handling and session control for AR conversational media at Gm reference point, such as the usage of SDP and setup and control of the individual media streams between clients, are defined in this specification.

NOTE 2: DC Application Repository may be in external DN but can also be in operator domain. The DC Application Repository holds the application(s) that can be used in AR communication sessions and is out of scope of 3GPP.

AR Application Server (AR AS):

- AR Application Server is responsible for AR service control related to AR communication, including AR session media control and AR media capability negotiation with the UE.

NOTE 3: AR Application Server is a specific DC Application Server and is out of scope of 3GPP.

NOTE 4: The UE can download the AR metadata from AR AS through application data channel.

DCSF:

- The DCSF receives event reports from the IMS AS, and decides whether AR communication service is allowed to be provided during the IMS session. Additionally, the DCSF interacts with the AR AS for DC resource control.

MF:

- Support AR conversational service by providing transcoding for terminals with limited capabilities. Additionally, the MF may collect spatial and media descriptions from UEs and create scene descriptions for symmetrical AR call experiences.

- Provide remote rendering for AR-MTSI clients in terminals with limited capabilities based on rendering negotiation. For remote rendering the AR-MTSI client provides AR metadata, e.g., pose data, as defined in clause 6 of this specification.

IMS AS:

- The IMS AS receives the media control instructions from the DCSF and accordingly interacts with the UE for connecting the UE's audio/video media termination to the MF [4], and interacts with MF for data channel media resource management for AR media processing.

\* \* \* \* Change #3 \* \* \* \*

## 6.1 General

Real-time scene creation for an AR conference with two or more participants may be done by the MF to create a symmetric experience for all participants. For an MF to create a scene, it may request the following information from the UEs:

- spatial description of the space surrounding the UE e.g., the occlusion-free space around the user in which the AR media will be rendered.

- media properties indicating the AR media that the UE will be sending, and thus have to be incorporated in the scene.

- receiving media capabilities of the UEs, which may include

- UE media decoding capabilities

- UE hardware capabilities (e.g., the display resolution)

- information based on detecting the location, orientation, and capabilities of physical world devices, eligible for usage in an audio-visual communications session

Based on this information the MF creates a scene which includes:

- defining the placement of the user and the AR media in that scene, including e.g., the position, size, depth from the user, anchor type, and recommended resolution (or quality)

- specific rendering properties for the AR media, e.g., for a 2D object to be rendered with a billboarding effect

The MF can then share the scene with the participant UEs using a supported scene description format. This scene description may be different for different UEs.

NOTE: The scene as sent by the MF allows the UE to 1) select and request any related media (for example, in a quality and bitrate based on the rendering characteristics or network connection), 2) render the complete scene on a (virtual) display device, and 3) update the rendering and requested media dynamically (e.g., according to the movement and view orientation of the user).

\* \* \* \* Change #4 \* \* \* \*

## 6.3 Spatial descriptions

### 6.3.1 Spatial description format

#### 6.3.1.1 General

A spatial description format is used for defining the physical space around a UE or trackable in which virtual content can be inserted. This clause includes the supported formats and the method for exchanging the information between AR-MTSI clients.

#### 6.3.1.2 Available visualization space

An AR-MTSI client in terminal may send available visualization space, user position and other trackable poses to AR MF for scene creation and update.

The available visualization space defines an occlusion-free space around the user for rendering the AR scene as a geometric primitive. The format for available visualization space is defined in clause 12.4 of TS 26.119 [3]. The type of the message containing visualization space as a payload shall be “**urn:3gpp:ar:v1:visualization-space**”. The availableVisualizationSpace object [3] shall contain a xrSpaceId. The xrSpaceId is used for determining the local coordinate axis of the visualization space. The xrSpaceId shall be a unique identifier for an XR space of one AR-MTSI client in terminal. If the visualization space is sent, then initial user pose shall be sent. The user pose and visualization space are in reference to the same xrSpaceId.

If the visualization space is anchored to another trackable (instead of the user) not anchored around the user or if the viewer is not the centre of the visualization space, an initial pose for a trackable as defined in clause 6.3.1.3 may be used.

\* \* \* \* Change #5 \* \* \* \*

## 6.4 Scene descriptions

An AR-MTSI client in terminal that is a compliant device type of TS 26.119 [3] shall support the capabilities requirements for scene description as described in clause 10 of TS 26.119 [3] for its respective device type.

When used in an AR call, the scene description should be the entry point to the AR session (after establishment of a regular call/conference) and shall be exchanged over the data channel as described in TS 26.114 [2]. The Scene Description is exchanged over a stream with a stream id in the range 1 to 1000 and shall be provided by the AR AS through the MF to the AR-MTSI client in terminal. In this case, no web application needs to be downloaded in this case.

Based on the information in the Scene Description, the UE may decide to add additional media streams through a re-INVITE. However, at least the RTP session for the voice of every participant should be present and should be linked to an audio source in the scene description.

NOTE: Support for advanced audio codecs, such as IVAS, in scene description is for further study.

Each participant should be associated with their own camera node, identified through the node name, which is also provided as part of the SDP through the “sd-nodes” attribute of media session of the data channel that carries the Scene Description.

The “sd-nodes” attribute shall conform to the following ABNF syntax:

att-field = "sd-nodes"

att-value = participant-label SP node-name \*("," node-name)

participant-label = char-val ; char-val is defined in RFC 7405

node-name = char-val ; char-val is defined in RFC 7405

The AR MF should apply pose updates from the received pose information of each participant to their respective camera nodes, as negotiated by the SDP sd-nodes attribute.

A scene description of an AR session may be sent from the AR MF/MRF to the AR-MTSI clients in terminal.

An AR MF that supports scene description shall support:

- The capability to generate a scene description file that conforms to the SD-Rendering-glTF-Core capability as defined in TS 26.119 [3].

- The capability to generate and update a scene description file that conforms to the SD-Rendering-glTF-Ext1 as specified in TS 26.119 [3].

An AR MF that supports scene description may additionally support the generation of scene description files and updates that conform to the **SD-Rendering-glTF-Ext2** capabilities as defined in TS 26.119 [3].

An AR MF that supports scene description may additionally support the generation of scene description files and updates that conform to the **SD-Rendering-glTF-Interactive** capabilities as defined in TS 26.119 [3].

In addition, an AR MF that supports scene description shall support the referencing of RTP streams in the scene description through the MPEG\_media extension as defined in ISO/IEC 23090-14 AMD 2 [7]. The external media shall be RTP media streams supported by an AR-MTSI client and signalled in the SDP.

When scene description is not used as the entry point, the scene description shall be sent by the AR MF/MRF to the AR-MTSI client in terminal over the application data channel. The type of the message containing the scene description shall be set to “**urn:3gpp:ar:v1:sd**”.

An AR MF that supports scene descriptions should create and distribute the scene for an AR call with audio and video streams based on the visualization space, viewer position and AR media properties. The AR MF should create the scene description for each participant (AR-MTSI client in terminal) such that the shared experience is symmetrical for the different users in the call, e.g., to maintain relative position of users and objects.

\* \* \* \* Change #6 \* \* \* \*

# 7 Media configurations

7.1 General

The media configuration requirements for MTSI clients in terminals specified in TS 26.114 [2], clause 6, also apply for AR-MTSI client in terminal.

An SDP framework for AR data exchange for AR communication is presented to negotiate codec support for AR media, AR metadata, as well as RTP/RTCP signalling necessary for AR media rendering processing.

AR-MTSI client in terminal shall use RTP for the real-time transport of AR media for AR communication. Any AR media as an overlay may refer to the overlay configuration described in clause Y.6.4.3 of TS 26.114 [2]*.*

AR-MTSI client in terminal shall use data channels for exchange of AR metadata and rendering negotiation. The SDP attribute “*3gpp\_armetadata\_types*”should be used to indicate the types of AR metadata which defined in clause 6 (e.g. pose, action and scene description) within the data channel.

The syntax for the SDP attribute is:

att-field = "3gpp\_armetadata\_types"

att-value = message\_type \*("," message\_type)

message\_type = char-val

 ; URN identifying the message type of AR metadata

 ; char-val is defined in RFC 7405

Poses as part of AR metadata may be transmitted via RTP session in a RTP header extension as specified in clause 4.3 of TS 26.522 [8].

7.2 Network media rendering configuration

The AR-MTSI client in terminal shall indicate its support for AR calls by including the “webrtc-datachannel” in the “+sip.sub-type” Contact header field.

A new Contact header field parameter, “+sip.3gpp-ar-support” is used to indicate the level of support for AR calls. The possible values for the “3gpp-ar-support” parameter are:

- “**ar-capable**”: indicates that the terminal is fully capable of receiving and rendering AR media as described by the capabilities in [2] clause 9.2.

- “**ar-assisted**”: indicates that the terminal is capable of transmitting AR metadata on the uplink. However, the UE has no support for processing and rendering a 3D scene. The participation in an AR call requires the deployment of network rendering. The rendered view(s) are controlled by the pose information that is shared by the terminal.

In the absence of the “+sip.3gpp-ar-support”, it shall be assumed that the terminal has no support for AR calls. In this case, the MTSI client can only participate in the AR call if network rendering is offered.

An AR-MTSI terminal that intends to participate in an AR call shall register with the “**ar-capable**” value for the “+sip.3gpp-ar-support” parameter and shall offer/answer an SDP that includes a data channel with the sub-protocol “mpeg-sd”. Any updates that the AR-MTSI terminal intends to share, including pose updates, will be sent as scene updates to the AR AS. An AR-MTSI terminal that intends to participate in an AR call with the support for network rendering shall register with the “**ar-assisted**” value for the “+sip.3gpp-ar-support” parameter and shall offer/answer an SDP that includes a data channel with the sub-protocol “3gpp-sr-metadata” as defined in [6]. Pose updates that are to be used for the rendering are shared as pose predictions with the MF.

As specified in Annex AC.9 of TS 23.228 [4], the AR application server may provide network assisted rendering. An AR-MTSI client in terminal can decide to request network media rendering based on user selection and its status such as power, signal, computing power, internal storage, etc. The AR-MTSI client in terminal shall complete an AR media rendering negotiation with the AR AS before it initiates subsequent procedures to activate the network media rendering. The data channel should be established for rendering negotiation with SDP offer/answer between AR-MTSI client in terminal and MF with the sub-protocol “3gpp-sr-conf”, and continue to be used for rendering re-negotiation until the end of the AR communication.

An **AR-assisted** terminal that intends to deploy network rendering for AR media rendering, shall use the negotiation processes between the AR-MTSI client in terminal and the AR AS to determine the split rendering configuration. The split rendering configuration shall be in JSON format as specified in clause 8.4.2 of TS 26.565 [6]. The exchange of the configuration information shall take place using the established application data channel. The split rendering configuration message shall be formatted according to clause 8.4.2.2 of TS 26.565 [6] and shall have the type: “**urn:3gpp:split-rendering:v1:configuration**”. The output description message shall be formatted according to clause C.1.4 of TS 26.565 [6] and shall have the type: “**urn:3gpp:split-rendering:v1:output**”.

For a terminal that does not support AR calls, the IMS AS may trigger network rendering on behalf of the terminal upon receiving an (re)INVITE for an AR call. The output format for the rendered media shall be conformant to clauses 10.4.3 and 10.4.4 of TS 26.119 [3]. The MF that performs the remote rendering shall select a suitable rendering viewpoint for the session, e.g. a selected viewpoint in the scene or the initial viewpoint for the participant as assigned by the AR AS in the scene description. In case no network rendering can be setup, the IMS AS should reject the call.

The IMS AS detects support for AR capabilities based on the presence or absence of the “+sip.3gpp-ar-support” parameter of the Contact Header Field in the REGISTER message.

\* \* \* \* Change #7 \* \* \* \*

Annex A (normative):
Call flows for IBACS

# A.1 IMS AR communication Call Flows

## A.1.1 General

Figure A.1.1-1 illustrates a high-level call flow for AR communication.


Figure A.1.1-1: High-level call flow for AR communication

The following steps are performed:

A. AR Call Session Setup: UE1 initiates an AR call, and the AR call session is established between UE1 and UE2, data channels along with audio and video channels are established between UE and network

B. Split Rendering Negotiation: When the UE has poor rendering capability which is not able to satisfy the requirements of AR communication, the split rendering negotiation is involved between the UE and IMS. The split may be adjustable during a session. This split rendering negotiation step can be executed/re-executed during the session when the UE’s status changed and/or user selection.

C. Scene Description Processing: Prepare and generate scene description updates for the AR call session, this procedure can be done either on UE or IMS network.

D. AR Media Processing: AR media & Metadata transmission and AR media rendering for the AR call session, if split rendering is enabled, this procedure can be done by both UE and IMS network.

## A.1.2 AR Call Session Setup

The AR call session procedure shall be followed as specified in clause AC.7 of TS 23.228 [4].

## A.1.3 Split Rendering Negotiation

Figure A.1.3-1 illustrates a detailed call flow for split rendering procedure.



Figure A.1.3-1: Call Flow for Split Rendering Negotiation

The steps are as follows:

1. The UE1 initiates an AR communication session and establishes audio and video session connections with the UE2. Then the bootstrap and application data channels are established for the UE1 and UE2.

2. When the UE1 discovers that its media capabilities cannot meet the AR communication related media rendering requirements, the UE1 decides to start split rendering call flow. Then the UE1 calculates which AR objects can be rendered by itself based on its status, and decides which part of the AR objects to be rendered in the UE1 and the others to be rendered in the IMS network.

3. The UE1 initiates the application data channels between the UE1 and the MF, for the split rendering request and metadata transmission.

4-5. The UE1 sends a Split Rendering Request to the MF through the established application data channel, the request includes the information of the AR objects to be rendered in IMS network, the MF transfers the request to the AR AS.

Note: The use of data channel for split rendering procedure (steps 2, 4 and 5) is optional. Alternatively, the MF or AR AS may decide to use split rendering for media delivery based on, e.g., device capability and media properties.

6. The AR AS may decide whether to provide AR media rendering function based on the request message received from the UE1, the media rendering resource available on the MF and the split rendering provisioning status for the UE1.

7. The AR AS sends a request to the DCSF to create required AR media rendering resource on the MF for the AR objects that should be rendered in IMS network, the DCSF transfers the request to the IMS AS.

NOTE: The requested AR media rendering resources are such that they can accommodate expected changes in scene description for the session.

8. The IMS AS sends the media resource allocation request to the MF, to reserve AR media rendering resource for the UE1.

9. When the MF resources are allocated successfully, the IMS AS returns a successful response to the DCSF, the DCSF transfers the response to the AR AS.

10-11 The AR AS returns a successful Split Rendering Response carrying the result to the MF. The MF then transfers the response/notification message with split rendering configuration information to the UE1 through the application data channel.

12. Subsequent procedures continue.

## A.1.4 Scene Description Processing

Figure A.1.4-1 illustrates a detailed call flow for scene description processing procedure.



Figure A.1.4-1: Call Flow for Scene Description Processing

The steps are as follows:

1. The UE1 initiates an AR communication session and establishes audio and video session connections with the UE2. Then the bootstrap and application data channels are established for the UE1 and UE2.

2. The split rendering negotiation procedure has been finished.

3. MF prepares the scene description based on media descriptions and assets for the call.

4. MF delivers the scene description to the UEs.

5. A UE may trigger a scene update e.g., when a new object is added/removed in the scene, or a spatial information update is sent or UE capabilities change. The figure shows the update is triggered by UE1, but this can be either UE.

6. The MF will process the new information and creates a scene description update. It is also possible for the MF to initiate an update without an update from the UEs, for example if link conditions change.

7. MF distributes scene description update to all UEs.

NOTE: Spatial data related updates may be required for collaborative AR calls, e.g., when multiple users are physically collocated and also part of the same AR experience. The type of spatial description updates is for further study.

8. Subsequent procedures continue.

## A.1.5 AR Media Processing

Figure A.1.5-1 illustrates a detailed call flow for AR media processing procedure.



Figure A.1.5-1: Call Flow for AR Media Processing

The steps are as follows:

1. The UE1 initiates an AR communication session and establishes audio and video session connections with the UE2. Then the bootstrap and application data channels are established for the UE1 and UE2.

2. The split rendering negotiation procedure has been finished.

3. The scene description processing has been finished.

4a. The UE1 captures the AR metadata, performs AR media rendering locally and then encodes rendered AR media, e.g., as audio/video media stream.

5a. The UE1 sends the rendered AR media and/or AR metadata to the peer through the established media connection(s).

6a. The UE1 receives the AR media and/or AR metadata from the peer through established media connection(s).

7a. The UE1 decodes the AR metadata and performs AR media rendering locally and displays it on its screen.

Network Centric Procedure:

4b. The UE1 retrieves AR metadata such as pose and user input locally, and renders the part of AR objects that should be done in the UE1 according to the result in step 2.

5b. The UE1 sends AR media to the MF, the AR media can include the audio/video media stream rendered by the UE1, and the AR metadata which is used for the AR objects rendering that should be done in the IMS network.

6b. The UE2 may also send AR media and/or AR metadata to the MF/MRF, e.g., the viewport.

7b. The MF renders the part of AR objects based on the AR metadata received. Then the MF decodes the audio/video media stream received from the UE1, and combines with the AR media rendered by the MF.

8b. The MF sends the rendered audio/video media stream to UE2.

9b. The MF sends the rendered audio/video media stream to the UE1.

10. A UE may trigger a scene update as described in clause 9.1.3. for example, if objects are added/removed from the scene or if UE capabilities change and it can no longer render some AR objects as negotiated in step 2.

11. The MF processes the new information and creates a scene description update. It is also possible for the MF to initiate an update without an update from the UEs, for example if network conditions change.

12. The MF distributes the scene description update to all UEs.

\* \* \* \* End of changes \* \* \* \*