**3GPP TSG-WG SA4 Meeting #133-e *S4-251538***

**Online, 18th - 25th July 2025**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.3* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **26.264** | **CR** |  | **rev** | **3** | **Current version:** | **19.0.0** |  |
|  | | | | | | | | |
| *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:*** | Draft base CR for AvCall-MED | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | Qualcomm | | | | | | | | | |
| ***Source to TSG:*** | S4 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | AvCall-MED | | | | |  | ***Date:*** | | | May 2025 |
|  |  | | | |  | |  | | |  |
| ***Category:*** | B |  | | | | | ***Release:*** | | | Rel-19 |
|  | *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 can be 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:*** | | AR Calls have been enhanced with Avatar communication capaibilities. The stage 3 aspects related to formats and signaling are missing. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | This CR adds the formats, protocols, and signaling to add support for Avatar communication to AR calls. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | Rel-19 will be missing stage 3 support for Avatar communication. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **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:*** | |  | | | | | | | | |

|  |
| --- |
| **1st Change** |

# 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 26.114: "IP Multimedia Subsystem (IMS); Multimedia telephony; Media handling and interaction".

[3] 3GPP TS 26.119: "Media Capabilities for Augmented Reality".

[4] 3GPP TS 23.228: "IP Multimedia Subsystem (IMS); Stage 2".

[5] 3GPP TS 24.229: "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3".

[6] 3GPP TS 26.565: "Split Rendering Media Service Enabler".

[7] ISO/IEC 23090-14 AMD 2, Information technology — Coded representation of immersive media — Part 14: Scene description — Amendment 2: Support for haptics, augmented reality, avatars, Interactivity, MPEG-I audio, and lighting

[8] 3GPP TS 26.522: "5G Real-time Media Transport Protocol Configurations".

[9] 3GPP TR 26.813, Avatar Representation and Communication (Release 19).

[10] 3GPP TS 23.228, IP Multimedia Subsystem (IMS); Stage 2 (Release 19).

[11] ISO/IEC 23090-39, Information technology — Coded representation of immersive media — Part 39: Avatar Representation Format.

|  |
| --- |
| **2nd Change** |

## 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].

**Animation data:** Skeletal, blend shape set, and other animation-related information.

**Animation stream:** Timed animation data sequence used to animate the base avatar.

**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.

**Asset:** An independently accessible component of an avatar.

**Avatar:** A digital representation of a user.

**Base avatar model:** Personalized and animatable model of the user.

**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]

|  |
| --- |
| **3rd Change** |

## 3.3 Abbreviations

|  |
| --- |
| **4th Change** |

## 4.3 End-to-End Reference Architecture

|  |
| --- |
| **5th Change** |

## 5.5 Avatars

### [Editor’s Note: Characterization information for the base avatar formats and animation data is expected.

### 5.6.1 General

An AR-MTSI Tx client in terminal offering an avatar in an AR Call shall support the Avatar Representation Format (ARF) as specified in [11] and shall have at least one base avatar stored in the Base Avatar Repository (BAR) in one of the two supported container formats defined in [11].

An AR-MTSI Rx client that supports avatars shall support both the ARF ISOBMFF and Zip container formats.

Editor’s Note: The ARF specification is still under development. Some details may change.

Editor’s Note: Profiles and levels for the required ARF features for 3D and 2D avatars may be defined in a new Annex in the specifiaction.

The base avatar shall comply with the ARF specification [11]. In addition, the ARF document shall include the following information:

- A list of the supported animations, which includes at least one animation type (e.g., face or landmark animation),

- At least one asset with at least one level of detail, and

- All data of relevant assets shall be contained in the ARF container of the base avatar.

### 5.6.2 3D Avatar Format

#### 5.6.2.1 General constraints

An AR-MTSI client that supports 3D avatars shall support the ARF base avatar format as specified in ISO/IEC 23090-39 [11] with the following specific requirements and with the simple profile constraints defined in 5.6.2.1.

#### 5.6.2.2 3D Avatar simple profile

[NOTE: The general constraints as well as the profile specific constraints stated in this clause are subject to justification prior to final approval.

Editor’s Note: general constraints will be listed in section 5.6.2.1

The 3D avatar container shall consist of the following mandatory components:

* At least one Skeleton component that defines the hierarchical joint structure for body animation, with support for at least partial humanoid joint configurations. Inverse Bind Matrices (IBMs), shall be provided for each joint of the skeleton,
* At least one Skin component that references both the skeleton and associated meshes to enable skeletal deformation through Linear Blend Skinning (LBS),
* 3D mesh geometry data that conforms to the binary glTF (GLB) format version 2.0 or later, with support for:
  + Vertex positions, normals, and texture coordinates
  + Triangle-based topology
  + Multiple levels of detail (LOD) for scalable rendering
* Skinning weight data provided as dense tensors in the format specified in Annex E of ISO/IEC 23090-39 [11], where:
  + Weights are represented as an NxM tensor (N vertices M joints)
  + Each vertex’s weights sum to 1.0 for proper deformation
  + Maximum of 4 joint influences per vertex for optimization
* Texture data components that conform to still image formats as defined in section 5.5, including:
  + Diffuse/albedo textures in Recommended normal maps for surface detail enhancement
  + Optional metallic-roughness textures for PBR rendering
  + Optional occlusion maps for ambient lighting
* For facial animation support, 3D avatars shall include:
  + At least one BlendshapeSet component that references the head/face mesh, containing:
    - A minimum of 50 blend shapes
    - Shape key data as meshes in GLB format, restricted to vertex positions, polygon/face information, normals, and tangents. All other information for the blend shape key is inherited from the base mesh.
  + At least one BlendshapeSet or a mapping table shall correspond to an OpenXR-specified blendshape set.

The ARF document for 3D avatars shall specify:

* Multiple animation types in the supportedAnimations list:
  + Body animation URNs for skeletal joint animation
  + Face animation URNs for blend shape animation
  + Optional hand animation URNs for detailed finger tracking
* A hierarchical asset structure supporting multiple levels of detail:
  + Each LOD shall maintain consistent UV mapping for texture compatibility
  + Lower LODs should preserve animation capability while reducing polygon count

The following performance and optimization recommendations apply:

* Mesh data should be optimized with:
  + Indexed geometry to reduce vertex duplication
  + Quantized vertex attributes where appropriate
  + Efficient texture atlas usage to minimize draw calls
* Data items of the 3D avatar should signal no compression or protection schemes by default to ensure broad compatibility.

NOTE: Compression aspects of ARF are for FFS.

NOTE: Content protection aspects are for FFS

* 3D avatars should target a maximum of 200,000 polygons for the highest LOD on mobile devices, with appropriate reductions for lower LODs.]

### 5.6.3 2D Avatar Format

NOTE: the general constraints as well as the profile specific constraints stated in this clause are subject to justification prior to final approval.

An AR-MTSI client that supports 2D avatars shall support the following data formats and requirements for the avatar assets:

* 2D mesh representations that consist of a single planar mesh or quad suitable for texture mapping, conforming to the binary glTF (GLB) format,
* Static image assets for the base avatar representation that conform to still image formats as defined in section 5.5,
* Landmark sets for facial animation, where landmarks shall be defined as 2D coordinates (x, y) on the avatar image plane,
* Texture data components that contain the avatar appearance, which shall conform to still image formats as defined in section 5.5,
* Animation data based on landmark positions, where the landmark animation samples shall use the 2D coordinate format (ala\_is\_3d\_flag set to false) as specified in clause 8.4 of ISO/IEC 23090-39 [11], and
* Sparse tensor data formats for landmark vertex indices as described in the ARF specification [11].

For 2D avatars, the ARF document shall include:

* At least one landmark set that defines facial feature points for animation,
* Support for the landmark animation type in the supportedAnimations list, identified by an appropriate URN,
* A simplified asset structure with a single level of detail containing the 2D mesh and associated texture, and
* LandmarkSet components that reference vertices on the 2D mesh corresponding to key facial features (e.g., eyes, mouth, eyebrows).

The 2D avatar animation should be achieved through landmark-based deformation, where:

* Landmark positions from the animation stream drive the deformation of the 2D mesh,
* The mesh deformation should preserve the natural appearance of the facial features, and
* The number of landmarks should be sufficient to capture essential facial expressions while maintaining computational efficiency.
* Data items of the 2D avatar should signal no compression or protection schemes to ensure broad compatibility.

Alternatively, support for animation using voice-based animation through a pre-trained and fine- tuned model for the user is possible. The DNN model is stored as part of the ARF container and declared as proprietary animation framework.

NOTE: The specific landmark configuration and animation URNs for 2D avatars may be defined in operator-specific profiles or through industry fora.

]

|  |
| --- |
| **6th Change** |

# 6 AR Metadata

## 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 information on whether each user should be represented by their avatar, and if so, the Avatar ID of the user, and the avatar capabilities of the UEs. 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).

An AR-MTSI terminal may request the MF to generate the animation streams based on the UE’s supplied media streams.

|  |
| --- |
| **7th Change** |

### 6.3.2 Avatar Animation Stream Format

[An AR-MTSI client or MF that supports avatars shall support the exchange of avatar animation data over the data channel according to the sample formats described in clause 8 of ISO/IEC 23090-39 [11].

NOTE: Support for other means to transport animation streams (e.g., over the media channel) may be added in the future.

When the data channel is used to send animation data, the metadata data channel message format defined in clause 6.2 shall be used and the avatar animation messages shall have the type “urn:3gpp:ar:v2:avatar:animation” and the format shown in Table 6.6-1.

Table 6.6-1: Message format for avatar animation messages

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Cardinality | Description |
| id | string | 1..1 | A unique identifier of the message in the scope of the data channel session. |
| type | string | 1..1 | urn:3gpp:ar:v2:avatar:animation |
| message | Object | 1..1 | Message content |
| subtype | number | 1..1 | An identifier of the subtype of the animation message. Value 1 indicates a facial animation, value 2 indicates a joint animation, and value 3 indicates a landmark animation. Other values are reserved for future use. |
| payload | Object | 1..1 | The avatar animation sample format corresponding to the message subtype. |

No compression scheme is defined for the animation samples.]

|  |
| --- |
| **8th Change** |

### 6.4.X Integration in Scene Description

|  |
| --- |
| **9th Change** |

### 6.5.4 Network Animation and Rendering

When an AR-MTSI client initiates or receives a call with avatar media, the IMS Application Server (IMS-AS) shall evaluate whether network-based avatar animation and rendering is required. The IMS-AS forwards avatar-related INVITE requests to the Avatar-capable AR AS for capability assessment and media routing decisions.

The AR AS shall invoke network-based animation and rendering through an MF when:

* the receiving MTSI client has not registered the "+sip.3gpp.avatar-animation" feature tag,
* the receiving MTSI client's registered capabilities indicate insufficient resources for avatar animation,
* the offered animation frameworks are not supported by the receiving MTSI client.

An AR-MTSI client that intends to use network-based animation and rendering (e.g. lacks the capability of animating and rendering the offered avatar), shall initiate the negotiation process between the AR-MTSI client and the AR AS to request avatar animation and rendering.

When network animation and rendering is invoked, the AR AS shall allocate an MF capable of real-time avatar rendering and configure it with the appropriate rendering parameters based on the receiving UE's video capabilities. The IMS AS shall modify the SDP to route avatar animation data to the MF instead of the receiving UE, effectively inserting the MF into the media path between the sending and receiving MTSI clients.

The sending AR-MTSI client shall establish a data channel with the MF for avatar animation parameters, while the MF shall establish a video stream with the receiving MTSI client using standard video codecs as specified in [2].

The MF performing network-based avatar animation and rendering shall fetch the ARF container from the BAR using the reference provided in the SDP and load the avatar model for rendering. The MF shall receive animation streams through the data channel from the sending AR-MTSI client and apply these animation samples to the avatar model in real-time. The animated avatar shall be rendered as a 2D video stream or stereoscopic 3D video stream based on the receiving UE's capabilities.

For the receiving MTSI client, the avatar data channel media description shall be replaced with a video media description including standard video codec negotiation as specified in [2], and avatar-specific attributes that are not applicable to video streams shall be removed.

The MF should ensure lip-sync between avatar animation and associated audio streams.

|  |
| --- |
| **10th Change** |

7.3 SDP Negotiation and Signaling of Avatars

|  |
| --- |
| **11th Change** |

8.4 Data Channel Transport of Avatar Data

|  |
| --- |
| **12th Change** |

## 9.2 Avatar-related QoE

### 9.2.1 Timing Information for Avatar Animation and Rendering

In avatar communication, compared to traditional remote or split rendering for AR, one important parameter to estimate the user quality of experience is the *posture to render to photon time delay,* defined as the time duration between the *posture-capture-time* or *animation-data-generation-time*, the *avatar-animation-time*, and the *actual-display-time*. The calculation or measurement of the timestamps related to this delay is dependent on the entity performing the avatar animation and rendering, which is also influenced by the latencies involved in generating and delivering the animation data required at the corresponding entity.

The *posture-capture-time* is measured in the sending UE, as the time when the sender UE user’s pose is captured in order to generate the animation data.

The *animation-data-generation-time* is measured either in the sending UE or the MF, as the time when the animation data is generated from the source data.

The *avatar-animation-time* is measured, depending on the animation mode, either in the sending UE, MF or receiving UE, as the time when the base avatar is animated and rendered.

The *actual-display-time* is measured in the receiving UE, as the time when then rendered avatar is displayed to the user.

These timestamps may be delivered to each corresponding entity via feedback messages in order to facilitate better quality of experience. Better QoE may be provided to the user either through an adjustment in pose correction (in the receiving UE), or by other means such as the re-negotiation of a more suitable entity for animation data generation and/or avatar animation and rendering.

|  |
| --- |
| **13th Change** |

A.2 Avatar Communication Call Flows

### A.2.1 General Avatar Call Flow



Figure A.2.1-1: IMS Avatar Delivery and Animation Flow

**Z. Base Avatar Generation Before Call Setup**

The base avatar is generated before step A. Call Setup and Capability Negotiation. It may be uploaded to the BAR by using the Avatar management interface defined in annex B.

**A. Call Setup and Capability Negotiation**

A.1 An audio/video session is established between UE1 and UE2 and parameters of the session are negotiated.

The list of Avatar ID(s) and/or Avatar Representations is downloaded to the UE by the following options:

* Pre-configured in the UE: The Avatar ID List and/or Avatar Representations is provisioned or downloaded to the UE before a data channel for avatar call is setup.,
* Through bootstrap data channel: The Avatar ID List is fetched by the DC AS from the BAR when the associated Avatar communication application is downloaded and transferred from the DC AS to the DCSF and downloaded to UE through bootstrap data channel (see details in Annex AC 11.3.1 in TS 23.228[2].
* Through application data channel: The Avatar ID List is fetched by the DC AS from the BAR and downloaded to the UE through application data channel).

Note: further details of avatar selection and negotiation are defined in clause A.2.3.

**B. Scene Description Retrieval**

The MF and the participating UEs retrieve the scene description. The scene description may be shared by the MF with the UEs, in case of a shared experience, or the UEs may have their own scene descriptions.

**C. Scene Description Update**

A scene update trigger occurs, e.g., if an object is added to or removed from a scene or if spatial information is updated. The update trigger may originate from the MF itself or the UEs. The UEs may update their scene descriptions independently or the MF may generate an updated scene description and share it with the UEs.

NOTE1: The step B and C are not needed for 2D avatar.

**D.1. Avatar Acquisition**

D.1.1: The MF loads the base avatar for UE1 from BAR.

**D.2. Avatar Delivery**

Alternative #1: Sender-centric

D.2a.1: The MF delivers the base avatar of UE 2 to UE1 through data channel.

Alternative #2: Receiver-centric

D.2b.1: The MF delivers the base avatar of UE1 to UE2 through data channel.

**D.3. Animation Data Generation**

Based on the capability negotiation result in step A, the UE or network may generate animation data.

Alternative #1: UE centric animation data generation

D.3a.1: The UE1 generates the animation data based on the source data (e.g., audio, video, text) or using an XR runtime. The animation data may be transformed from the source data (e.g., from audio to text), or the same as the source data.

D.3a.2: UE1 delivers the animation data to the entity actuating avatar animation through data channel. The animating entity may be the MF or UE2.

Alternative #2: Network centric animation data generation

D.3b.1: UE1 sends source data for animation data generation to the MF over RTP (audio, video, text) or data channel (text).

D.3b.2: The MF processes the received source data to generate animation data during the session. The animation data may be transformed from the source data (e.g., from audio to text, video to motion data), or the same as the source data.

D.3b.3: The MF delivers animation data over data channel to the UE2 animating the base avatar. If network centric avatar animation is used, this step will be skipped. The animation data may be delivered to UE1 as well.

**D.4. Avatar Animation**

Based on the capability negotiation result in step A, the UE or network may animate the avatar.

Alternative #1a: Sender-centric avatar animation

[Optional] D.4a.1: UE2 delivers its pose information to UE1 for viewer-dependent avatar animation and rendering.

D.4a.2: UE1 animates and renders the base avatar using animation data. The animation data is generated by UE1 in step D.3a.1.1

D.4a.3: UE1 delivers the animated and rendered avatar to UE2. The animated and rendered avatar may be delivered as a 2D video through RTP.

D.4a.4: UE2 corrects the rendered video (for latency compensation) from UE1 before displaying as rendered avatar.

D.4a.5: UE2 delivers a report of timing information, including its actual display time to UE1 for the monitoring of the UE1 centric rendering service.

Alternative #1b: Receiver-centric avatar animation

D.4b.1: UE2 animates and renders the base avatar using animation data. The animation data may be generated by the MF, following steps D.3b.1 to D.3b.2 and received by UE2 in step D.3b.3 or it may be generated by UE1 in step D.3a.1 and received by UE2 in step D.3a.2.

D.4b.2: UE2 delivers a report of timing information, including its actual display time to UE1 (and MF) for the monitoring of the UE1 centric rendering service.

Alternative #1c: Network-centric avatar animation

D.4c.1: The MF animates and renders the UE1’s base avatar using animation data. The animation data may be generated by the MF, following step D.3b.1 and D.3b.2 or it may be received from UE1 following steps D.3a.1 and D.3a.2.

D.4c.2: The MF delivers the animated and rendered avatar to the UEs. In the figure, delivery to UE2 is shown as example. The animated and rendered avatar may be delivered as a 2D video through RTP.

D.4c.3: UE2 corrects the rendered video (for latency compensation) from the MF before displaying as rendered avatar.

D.4c.4: UE2 delivers a report of timing information, including its actual display time to UE1 (and MF) for the monitoring of the network-centric rendering service.

NOTE2: Rendering is not needed for 2D avatar.

### A.2.2 Avatar Management Call Flow



Figure A.2.2-1: Avatar management call flow via IMS network for registering and uploading base avatar and associated assets

Figure A.2.2-1 depicts the call flow procedure for registering and uploading a user’s base avatar and associated assets. The main steps in the call flow are as follows:

1. The UE creates the base avatar.

2. Base avatar registration (the use of an Avatar ID assigned by the BAR is required for the secure upload of the base avatar by the UE):

2.1 The UE sends registration request to the MF/DC AS via application data channel to request the registration of its base avatar.

2.2 The DC AS forwards the registration request to the BAR.

2.3 The BAR assigns a unique Avatar ID and URL for the base avatar of the UE according to the UE identifier known via the DC AS.

Note: How the BAR obtains and maps user identifiers to Avatar IDs is FFS.

2.4 The BAR sends a registration response containing the Avatar ID and URL for the registered base avatar to the UE via the MF/DC AS.

3. The UE associates the assigned Avatar ID to the Base Avatar data created from step 1.

4. The UE uploads the base avatar data to the BAR via the MF using the application data channel.

5. The BAR replies with associated information for the registered base avatar.

6. Adding new assets to the base avatar:

6.1 The UE creates or obtains a new asset for adding to the base avatar.

6.2 The UE sends an asset registration request for the new asset to the BAR via the MF using the application data channel.

6.3 The BAR creates a new Asset ID for the new asset and associates it to the Avatar ID of the base avatar.

6.4 The BAR updates the associated information corresponding to the Avatar ID.

6.5 The BAR sends an asset registration response containing the Asset ID to the UE via the MF.

6.6 The UE uploads the asset data to the BAR via the MF.

6.7 The BAR replies with updated associated information.

6.8 Steps 6.1 to 6.7 are repeated for the registration of additional assets.

### A.2.3 Avatar Selection and Negotiation Call Flow

For avatar communication over the IMS data channel, the avatar ID list (a list of the base avatars available in the BAR) is obtained by the UE using one of following options:

- Pre-configured in the UE: The Avatar ID List and/or Avatar Representations are provisioned or downloaded to the UE before any session for the avatar call is established.

- Through bootstrap data channel: The Avatar ID List is fetched by the DC AS from the BAR when the associated Avatar communication application is downloaded and transferred from the DC AS to the DCSF and downloaded to UE through the bootstrap data channel.

- Through application data channel: The Avatar ID List is fetched by the DC AS from the BAR and downloaded to the UE through an application data channel.

Three avatar animation modes are defined for avatar communication over the IMS data channel:

- Sender-centric: the sender UE animates and renders its base avatar before sending it to the receiving UE as 2D video.

- Receiver-centric: the receiving UE animates and renders the sender UE’s base avatar.

- Network-centric: the MF animates and renders the sender UE’s base avatar before sending it to the receiving UE as 2D video.

The decision of which avatar animation mode to use for avatar communication is dependent on the outcome of the capability and Avatar Animation Negotiation procedure via an application data channel, as detailed in figure A.2.3-1.



A.2.3-1: Avatar selection and negotiation call flow

The avatar animation negotiation procedure is based on the avatar type (2D or 3D) and the capability information of the sender/receiver UEs and the MF. The capability information includes the animation data type(s) (e.g., text, expression data, and motion signals for joints) supported by either UEs or the MF. For network centric mode, after avatar animation negotiation, the IMS AS instructs the MF to download UE1’s base avatar from the BAR, generate animation data from the source data received from UE1, and animate UE1’s base avatar using the animation data received from UE1 or generated by the MF itself.

A.1.0: (optional) An Avatar ID List is pre-downloaded, or pre-configured in UE1.

NOTE: Step A.1.0 is optional; in this step the Avatar ID List is provisioned or downloaded to the UE before any session for the avatar call is setup. The UE and the BAR may interact by means out of the scope of 3GPP.

A.1.1: An IMS session is established between UE1 and UE2, and a bootstrap data channel is established between UE1, the MF, and the DC AS.

A.1.2: The Avatar ID List and the Avatar Communication App are downloaded to UE1 via the BDC (see details in AC 11.3.1 in TS 23.228 [X]).

A.1.3: A P2A2P application data channel for Avatar Animation Negotiation is established between UE1, the MF DC AS, and UE2. If the Avatar ID List has not been downloaded in either steps A.1.0 or A.1.2, it may optionally be requested via this ADC.

A.1.4.1. (optional): If an Avatar ID list is not obtained in A.1.0 or A.1.2, UE1 may send an Avatar ID List request to the DC AS via MF; the DC AS then sends the request to the BAR.

A.1.4.2 (optional) If the BAR receives an Avatar ID List request, the BAR (generates and) sends the Avatar ID List to UE1 via the DC AS and MF

NOTE: Steps A.1.4.1 and A.1.4.2 are optional. Whether and which user identity(ies) should be used by the user of the sending UE (UE1) and/or the receiving UE (UE2) for the download of the avatar representations in the case of a receiving UE centric rendering mode will be decided by SA WG3 and the procedure will be aligned with SA WG3’s decision.

A.1.5: UE1 selects an avatar representation to be used for the avatar call using the list of available avatar representations known to UE1 via the Avatar ID List.

A.1.6: Avatar animation negotiation takes place via the established P2A2P ADC.

A.1.6.1: UE1 sends an avatar animation negotiation request using the ADC through the MF to the DC AS. The message carries parameters which may include: an avatar ID associated with the selected avatar representation selected in step A.1.4, animation data types (e.g., text, expression data, or motion signals for joints) supported by UE1, and related rendering requirements or capability information.

A.1.6.2: (optional) To facilitate the negotiation of the rendering mode, the DC AS may interact with the current serving MF to check the MF capability through the DC1, DC2, and DC3/DC4 interfaces The MF sends a response with its avatar capability information to the DC AS that enable the DC AS to make further decisions.For example, to determine whether network-centric rendering is supported and which rendering mode (MF or DC AS) to use.

Editor’s Note : Whether the interaction between the DC AS and the MF is supported needs confirmation from SA2. In particular, the MF response with its supported capabilities to the DC AS.

A.1.6.2: (optional) Through an established P2A application data channel, MF/DC AS sends a capability negotiation request to UE2. The message may include the same information as in described in A.1.5.1.

A.1.6.3: (optional) UE2 sends a capability negotiation response to the MF/DC AS. The message carries the capability negotiation result related to UE2’s preference.

A.1.6.4: The DC AS gets the avatar type (2D or 3D) associated with the avatar ID from base avatar retrieved from BAR or to be generated by the MF and confirms the avatar animation negotiation result based on the avatar type and the capabilities supported by UE1, the MF, and UE2. The capability negotiation result includes the rendering mode and animation method (e.g., by audio, text, or expression data and motion signals for joints).

A.1.6.5: The DC AS sends the capability negotiation response to UE1 through the MF. The message carries the capability negotiation result.

|  |
| --- |
| **14th Change** |

### A.2.4 ADC and RTP Stream Establishment for Avatar Call

As a result of the avatar selection and negotiation procedure as described in clause A.2.3, depending on the configuration negotiated, certain application data channels and/or RTP streams are established between different endpoints (UE1, MF, UE2) for the delivery of different avatar data components as shown in figure A.2.1-1. These include (where necessary) the delivery of scene description, base avatar, animation source data, animation data, pose information and feedback information.

The decision of which avatar animation mode to use for avatar communication is dependent on the outcome of the capability and Avatar Animation Negotiation procedure, as detailed in figure A.2.3-1.



A.2.4-1: ADC(s) and RTP stream establishment call flow

Avatar selection and avatar animation mode negotiation takes place as described in clause A.2.3. Depending on the configuration negotiated, delivery channels are established between certain endpoints as shown in figure A.2.4-1.

A.1.6: Depending on capability negotiation result for the selected avatar representation, media re-negotiation takes place in order to establish the necessary ADCs and RTP streams to deliver the various avatar data. In the case where media processing in the network is required but not supported by the allocated MF, an additional MF supporting such processing may be allocated. Avatar data may include: scene description, base avatar, animation data, pose information, and feedback information.

Sending UE Centric:

A.1.6S.1: P2A ADC (UE1) is established for base avatar delivery (figure A.2.1-1 step D.2a.1).

A.1.6S.2: P2A2P ADC is established for scene description retrieval and update (figure A.2.1-1 steps B and C).

A.1.6S.3: P2P ADC is established for pose information and/or feedback information (figure A.2.1-1 steps D.4a.1 and D.4a.5).

Receiving UE Centric:

A.1.6R.1: P2A ADC (UE2) is established for base avatar delivery (figure A.2.1-1 step D.2b.1).

A.1.6R.2: P2A2P ADC is established for scene description retrieval and update (figure A.2.1-1 steps B and C).

*alt#1a Animation Data Generation in sending UE*

A.1.6R.3a: P2P ADC is established for avatar animation data delivery (figure A.2.1-1 step D.3a.2).

*alt#1b Animation Data Generation in Network*

A.1.6R.3b: P2A ADC is established for animation source data delivery (figure A.2.1-1 step D.3b.1).

A.1.6R.4b: P2A ADC (UE2) is established for avatar animation data delivery (figure A.2.1-1 step D.3b.2).

A.1.6R.5: P2P ADC is established for feedback information (figure A.2.1-1 step D.4b.2)

Network Centric:

A.1.6N.1: P2A2P ADC is established for scene description retrieval and update (figure A.2.1-1 steps B and C).

*alt#1a Animation Data Generation in Network*

A.1.6N.2a: P2A ADC is established for animation source data delivery (figure A.2.1-1 step D.3b.1).

*alt#1b Animation Data Generation in sending UE*

A.1.6N.2b: P2A ADC is established for avatar animation data delivery (figure A.2.1-1 step D.3a.2).

A.1.6N.3: P2P ADC is established for feedback information (figure A.2.1-1 step D.4c.4)

|  |
| --- |
| **15th Change** |

Annex B:  
Base Avatar Management Interface   
(Informative)

## B.1 Mbar\_Management service

### B.1.1 Overview

This clause defines the BAR management API offered by the BAR and used by the DC AS or MF to manage avatar related data in the BAR. A summary of the resource structure is shown in table B.2.2-1 below.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **HTTP request path element hierarchy** | **Description** | **Allowed HTTP methods** | | | | | **Resource** | **OpenAPI** |
| **Create** | **Retrieve** | **Update** | **Destroy** | **Non-RESTful operation** | **structure definition clause** | **definition clause** |
| avatars | Avatar collection | POST |  |  |  |  | B.1.2 | X.X.X |
| *{avatarId}* | Avatar resource |  | GET | PUT | DELETE |  |
| assets | Asset collection | POST |  |  |  |  | B.1.3 | X.X.X |
| *{assetId}* | Asset resource |  | GET | PUT | DELETE |  |
| associatedInfo | Associated Information resource |  | GET |  |  |  | B.1.4 | X.X.X |

### B.1.2 Avatars API

#### B.1.2.1 Overview

The Avatars API is used by the DC AS or MF to manage Base Avatars (including related assets and associated information) in the BAR, providing operational functions such as Base Avatar creation, retrieval, update and deletion.

#### B.1.2.2 Resource structure

The Avatars API is accessible through the following URL base path:

{apiRoot}/3gpp-mbar-management/{apiVersion}/avatars/

Table B.1.2.2-1 specifies the operations and the corrresopnding HTTP methods that are supported by this API. In each case, the sub-resource path specified in the second column of the table shall be appended to the above URL base path.

Table B.1.2.2-1: Operations supported by the Avatars API

|  |  |  |  |
| --- | --- | --- | --- |
| Operation name | Sub‑resource path | Allowed HTTP method(s) | Description |
| Create Avatar |  | POST | Creates a new avatar resource in the BAR. |
| Get Avatar | {avatarId} | GET | Used to retrieve a previously created or uploaded base avatar in the BAR. |
| Update Avatar | PUT | Used to upload or update Base Avatar data corresponding to an Avatar ID. |
| Delete Avatar | DELETE | Removes and deletes a Base Avatar, as well as its related assets and associated information. |

#### B.1.2.3 Data model

##### B.1.2.3.1 Avatar resource

| Property name | Data type | Cardinality | Description |
| --- | --- | --- | --- |
| avatarId | ResourceId | 1..1 | A unique identifier assigned to a Base Avatar by the BAR on creation. |
| assetIds | array(ResourceId) | 0..1 | A list of assets associated with the Base Avatar. |
| avatarContainer | URL | 0..1 | Payload containing the Base Avatar data and associated assets. This provides access to the full binary avatar container, including all of the contained assets. For creation and update operations, the URL shall point to a multi-part mime part with MIME type “model/vnd.mpeg.arf+zip”. |
| associatedInfo | AssociatedInfo | 0..1 | Associated information related to the Base Avatar. |

Table A.X.2.1.2.1-1: Definition of Avatar resource

### B.1.3 Assets API

#### B.1.3.1 Overview

The Assets API is used by the DC AS or MF to manage individual assets of the base avatar in the BAR, providing operational functions such as asset creation, retrieval, update and deletion.

#### B.1.3.2 Resource structure

The Assets API is accessible through the following URL base path:

{apiRoot}/3gpp-mbar-management/{apiVersion}/avatars/{avatarId}

Table B.1.3.2-1 specifies the operations and the corrresopnding HTTP methods that are supported by this API. In each case, the sub-resource path specified in the second column of the table shall be appended to the above URL base path.

Table B.1.3.2-1: Operations supported by the Assets API

|  |  |  |  |
| --- | --- | --- | --- |
| Operation name | Sub‑resource path | Allowed HTTP method(s) | Description |
| Create Asset |  | POST | Creates a new asset resource in the BAR. |
| Get Asset | {assetId} | GET | Used to retrieve a previously created or uploaded asset in the BAR. |
| Update Asset | PUT | Used to upload or update asset data corresponding to an Asset ID. |
| Delete Asset | DELETE | Removes and deletes an asset. |

#### B.1.3.3 Data model

##### B.1.3.3.1 Asset resource

Table A.X.2.1.2.1-1: Definition of Asset resource

| Property name | Data type | Cardinality | Description |
| --- | --- | --- | --- |
| assetId | ResourceId | 1..1 | A unique identifier assigned to an asset by the BAR on creation. |
| namespace | string | 1..1 | A namespace defining the intended usage of the asset, as exemplified by names such as "human/head" or "accessory/hat" |
| LoD | array(string) | 0..1 | A list of available LoDs for the corresponding asset. |
| assetData | array(URL) | 0..1 | List of URLs that point to the asset data. The primary URL shall point into an ARF document that describes all components of the asset. For creation/update of an asset, all components shall be provided as part of a multi-part mime body. |
| associatedInfo | AssociatedInfo | 0..1 | Associated information related to the Base Avatar. |
| Editor’s NOTE: The labels for LoDs require further discussions. | | | |

### B.1.4 Associated Information API

#### B.1.4.1 Overview

The Associated Information API is used by the DC AS or MF to fetch Associated Information related to a Base Avatar from the BAR.

#### B.1.4.2 Resource structure

The Associated Information API is accessible through the following URL base path:

{apiRoot}/3gpp-mbar-management/{apiVersion}/avatars/{avatarId}

Table B.1.4.2-1 specifies the operations and the corrresopnding HTTP methods that are supported by this API. In each case, the sub-resource path specified in the second column of the table shall be appended to the above URL base path.

Table B.1.4.2-1: Operations supported by the Avatar API

|  |  |  |  |
| --- | --- | --- | --- |
| Operation name | Sub‑resource path | Allowed HTTP method(s) | Description |
| Get Associated Information |  | GET | Used to retrieve associated information corresponding to a Base Avatar (identified by the AvatarID) in the BAR. |

### B.1.4.3 Data model

### B.1.4.3.1 Associated information resource

Table B.1.4.3.1-1: Definition of Avatar resource

| Property name | | | | Data type | | Cardinality | | Description | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| associatedInfo | | | | Object | | 1..1 | | A list of assets associated with the Base Avatar. | |
|  | avatarId | | ResourceId | | 1..1 | | A unique identifier assigned to a Base Avatar by the BAR on creation. | |
|  | avatarMetada | | Object | | 1..1 | | Metadata related to the Avatar, | |
|  | assetIds | | array(ResourceId) | | 1..1 | | A list of assets associated with the Base Avatar. | |
|  |  | assetLoDs | array(Object) | | 1..1 | | A list of available LoDs for the corresponding asset.  The resulting size in bytes shall be associated with each LoD. | |
|  |  | selectionInfo | Object | | 0..1 | | Provides information that the user can use to select this avatar. This may contain a name, a nickname of the asset, usage context e.g. casual, work, and images of renditions of the asset. | |
|  | supportedAnimations | | array(string) | | 1..1 | | A list of the URNs that identify the supported animation frameworks by this base avatar. | |
|  | infoUpdatedAt | | number | | 1..1 | | A timestamp (in wall clock time) describing the time of the last update to the associated information for the corresponding Base Avatar. This field is updated whenever an asset is modified by the owner of the Base Avatar or BAR. Users may utilize the infoUpdatedAt field to verify the latest validity of previously downloaded Base Avatar data. Comparing the infoUpdatedAt value with the downloaded time allows users to determine if the downloaded data requires updating, | |
| Editor’s NOTE: LODs descriptions/labels and their associated complexity require further discussion.  Editor’s NOTE: clarification about how LoD would be described and what that would mean in terms of complexity is expected. | | | | | | | | | |