**3GPP TSG-SA WG4 Meeting #124 S4-230854 is revised to S4-231021**

**Meeting, 22 – 26 May 2023**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.2* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **26.806** | **pCR** |  | **rev** | **-** | **Current version:** | **1.2.0** |  |
|  | | | | | | | | |
| *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:*** | Call flows for standalone AR glasses-based device architecture | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | Nokia | | | | | | | | | |
| ***Source to TSG:*** | S4 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | FS\_SmarTAR | | | | |  | ***Date:*** | | | 16-05-2023 |
|  |  | | | |  | |  | | |  |
| ***Category:*** | B |  | | | | | ***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 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-16 (Release 16) Rel-17 (Release 17) Rel-18 (Release 18) Rel-19 (Release 19)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | The latest version of TR 26.806 v 1.2.0 has incomplete call flows under clause 5.2. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | This CR proposes new call flows under clause 5.2.  Three new sub-clauses 5.2.1, 5.2., 5.2.3 are created in order to be more clear. Under each sub-clauses respective call flows which are left incomplete are newly added with corresponding text. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | clause 5.2 will remain incmplete leading to mis-interpretation of text. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 5.2, 5.2.1 (new), 5.2.2 (new), 5.2.3 (new). | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **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 |

## 5.2 Call flows

### 5.2.1 Call flows for standalone AR glasses-based device architecture

For the network architecture for the standalone AR glasses-based device architecture as shown in Figure 5.2.1-1, the call flow is provided in Figure 5.2..1-1, which is similar to the call flow in subclause 4.3.1 of TR 26.998.

Note that the 5G device serves as a relay, and may optionally be offloaded to perform some "phone based processing functions".

The call flow is described below.

1. The AR glasses device and the 5G device/phone sets up the tethering link.

2. The application contacts the application provider to fetch the entry point for the content. The acquisition of the entry point may be performed in different ways and is considered out of scope. An entry point may for example be a URL to a scene description.

3. Session set up:

3a. In case when the entry point is a URL of a scene description, the application initializes the Scene Manager using the acquired entry point.

3b. The Scene Manager retrieves the scene description from the scene provider based on the entry point information.

3c. The Scene Manager parses the entry point and creates the immersive scene.

3d. The Scene Manager requests the creation of a local AR/MR session from the XR Runtime.

3e. The XR Runtime creates a local AR/MR session and performs registration with the local environment.

Then steps 4 and 5 run in parallel:

4: XR Media Delivery Pipeline: In case when entry point is a scene URL, a delivery session - for accessing scenes (new scenes or scene updates) and related media over the network is established. This can basically use the MAF as well as the scene manager and the corresponding network functions. Details are introduced in Figure 5.2.1-2.

NOTE: The realization of XR media delivery pipeline may vary in different architectures.

5: XR Spatial Compute Pipeline: A pipeline that uses sensor data to provide an understanding of the physical space surrounding the device to determine the device’s position and orientation and placement of AR objects in reference to the real world and uses XR Spatial Description information from the network to support this process. Details are introduced in Figure 5.2.1-3.

6: Steps 4 and 5 run independently, but the results of both pipelines (e.g., media organized in a scene graph and pose of the AR device) are inputs of the AR/MR Scene Manager function. This function handles the common processing of the two asynchronous pipelines to create an XR experience.



Figure 5.2.1-1: Call flow for the tethering architecture standalone AR glasses-based device architecture.

The XR media delivery pipeline (step 4 of Figure 5.2.1.-1) in Figure 5.2.1-1 is provided in Figure 5.2.1-2, which is similar to the media delivery pipeline in subclause 4.3.2 of TR 26.998.

For an XR Media Delivery Pipeline:

1. The Scene Manager initializes XR scene delivery session.

2. The MAF establishes XR scene delivery session.

3. The MAF may receive updates to the scene description from the scene provider

4. The MAF passes the scene update to the Scene Manager.

5. The Scene Manager updates the current scene.

6. The Scene Manager acquires the latest pose information and the user’s actions

6. The Scene Manager in the device shares that information with the Scene Manager in edge/cloud

The media rendering loop consists of the following steps. Note that steps 8, 9 and 10 are running as 3 parallel loops:

8. For each new object in the scene:

a. The Scene Manager triggers the MAF to fetch the related media.

b. The MAF creates a dedicated media pipeline to process the input.

c. The MAF establishes a transport session for each component of the media object.

9. For each transport session:

a. The media pipeline fetches the media data. It could be static, segmented, or real-time media streams.

b. The media pipeline processes the media and makes it available in buffers.

10. For each object to be rendered:

a. The Scene Manager gets processed media data from the media pipeline buffers

b. The Scene Manager reconstructs and renders the object

11. The Scene Manager passes the rendered frame to the XR Runtime for display on the tethered standalone AR glass-based device.



Figure 5.2.1-2: Media delivery pipeline for call flow in Figure 5.2.1.

The XR spatial compute pipeline (step 5 of Figure 5.2.1) in Figure 5.2.1 is provided in Figure 5.2 1-3 (below), which is similar to the XR spatial compute pipeline in subclause 4.3.3 of TR 26.998.

Figure 5.2.1-3 Functional diagram for XR Spatial Compute Pipeline for call flow in Figure 5.2.1

For a XR Spatial Compute downlink delivery session:

1. The XR Spatial Compute function in the XR Runtime asks the MAF to establish a XR Spatial Compute downlink delivery session

2. The MAF communicates with the network to establish the proper resources and QoS

3. The XR Spatial Compute function requests access to XR Spatial Description information

4. An XR Spatial Description downlink delivery session is established across the XR Spatial Compute server, the media delivery function, the media access function and XR Spatial Compute function on the device.

5. XR Spatial Description information is delivered in this downlink delivery session

For a XR Spatial Compute uplink delivery session:

6. The XR Spatial Compute function in the XR Runtime asks the MAF to establish a XR Spatial Compute uplink delivery session

7. The MAF communicates with the network to establish the proper resources and QoS

8. The MAF established an appropriate uplink delivery pipeline

9. An XR Spatial Description uplink delivery session is established across the XR Spatial Compute function on the device, the media access function, the media delivery function and the XR Spatial Compute server.

10. Spatial compute information is upstreamed to the XR Spatial Compute server.

11. Data is continuously exchanged between the Scene Manager and the XR Runtime

|  |
| --- |
| End of change |

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

### 5.2.2 Call flows for display AR glasses-based device architecture without edge rendering

For the network architecture corresponding to the display AR glasses-based device architecture without edge rendering, as shown in Figure 5.2.-2, the call flow is shown in Figure 5.2.1.-3. The call flow is different from the one in Figure 5.2.1.-1 mainly in that the AR/MR Application, Scene Manager and Media Access Function reside on the 5G Device. Additionally, an XR Runtime API is on the 5G Device, the XR Runtime is on the AR Glasses Device, and the interactions between the XR Runtime API and the XR Runtime are proprietary. This simplifies the design from the point of view of the AR/MR application because it only needs to concern about the XR Runtime API.

The call flow is similar to the one in Figure 5.2.1-3 and the main difference is on the procedures related to XR Runtime. For example, step 3d: Establish AR/MR session now points to XR Runtime API, instead of XR Runtime.



Figure 5.2.1-1-: Call flow for the network architecture for tethered display AR glasses-based device without edge rendering.

The XR media delivery pipeline (step 4 of Figure 5.2.-3) in Figure 5.2-3 is provided in Figure 5.2-4.



Figure 5.2.2-2-: Media delivery pipeline for call flow in Figure 5.2-3.

The XR spatial compute pipeline (step 5 of Figure 5.2-3) in Figure 5.2-3 is provided in Figure 5.2-x below.



Figure 5.2.2-3 Call flow for XR spatial compute for for call flow in Figure 5.2-3.

|  |
| --- |
| End of change |

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

### 5.2.3 Call flows for tethered AR glasses with 5G relay with edge rendering

For the network architecture corresponding to the tethered AR glasses with 5G relay with edge rendering, as shown in Figure 4.2.2 2, the call flow is shown in Figure 5.2.3-1.

The call flow is different from the one in Figure 5.2-1 and Figure 5.2.3 mainly in that the XR Scene Manager, XR Runtime Functions are located within the AR Glasses and the Cloud/Edge. An XR Runtime API is located in the Cloud/Edge.

The call flow is similar to the one in Figure 5.2-3 and the main difference is on the procedures related to XR Runtime. For example, step 3, the XR link initialization and Step 4: session set-up takes place between XR Runtime and XR Runtime API. Establish AR/MR session now points to XR Runtime API, instead of XR Runtime. Step 5 and Step 6 follows similar procedure as the previous architectures.



Figure 5.2.3-1 Call flow for the network architecture for Tethered AR glasses with 5G relay.

|  |
| --- |
| End of change |