3GPP TSG SA WG4#123-e ***Tdoc S4-230606r1***

Online, 17th – 21st April 2023 revision of S4-230606

**Source: Fraunhofer IIS**

**Title: On ISAR Requirements**

**Document for: Discussion & Agreement**

**Agenda Item: 7.9**

## 1. Introduction

At SA4#122 the ISAR Work Item [1] was agreed and since then approved by TSG SA. This contribution aims at phase 1 of the ISAR WI, to […] identify and agree relevant requirements to be documented in a TR. This shall cover:

* Design constraints related to complexity and memory as well as constraints related to relevant interfaces between presentation engine and end device such as bit rate, latency, down- and upstream traffic characteristics.
* Design constraints related to functional capability requirements such as rendering of non-diegetic sounds, 3DoF rendering of diegetic immersive sounds, 6DoF rendering of diegetic immersive sounds, including simultaneous rendering of different sound categories, and room acoustics synthesis.
* Performance requirements.

## 2. On “Interfaces” for split rendering

Since the time the WI got approved and the time of writing this document, further discussions hinted at an issue that the cut between presentation engine and XR runtime may be misguiding for a discussion on the interfaces. The interfaces at this cutting point are for the typically APIs such as OpenXR, OpenGL, Vulkan, WebXR, engine-specific APIs, etc. For the visuals and OpenAL, WebAudio, or platform-specific APIs such as CoreAudio, ALSA, etc. For audio. Those APIs are typically in-device memory interfaces with low-latency, large data throughput.

Since those APIs are generally outside 3GPP’s scope, a split limited to `between presentation engine and XR runtime` is debateable, as the WI specifically aims to use a 5G connection between the two elements of a split rendering scenario.

The source thus believes that for the requirements aspect of the ISAR WI rather the interfaces as currently described in the MeCAR context would be relevant and one should look at the different scenarios where a split rendering (potentially only for the visuals where complexity is a significant burden) is assumed. The current TR26.998 and TR26.806 provide some guidance on such scenarios, where the split can happen between the AR Glasses device, 5G device/phone, and the cloud/edge, with multiple options whether a particular element is a 5G UE or AS or also a device connected by other than 3GPP means. Thus, it is suggested to create an overview of all such cases and the relevant interface points.

To come back at the definition of Split AR/MR, TR26.998 provides guidance:

- Split: the tethered device (phone/puck) or external entity (cloud/edge) does some power-intense processing (e.g., a pre-rendering of the viewport based on sensor and pose information), and the AR/MR device and/or tethered device performs post-processing considering the latest sensor information (e.g. warping to apply pose correction). Different degrees of split workflow exist, between different devices and entities. Similarly, vision engine functionalities and other AR/MR functions (such as AR/MR media reconstruction, encoding and decoding) may be subject to split computation.

Given the different scenarios and different degrees of split workflows and the ongoing parallel work e.g. in FS\_SmartAR and also other WIs in SA4 scope, the source suggests a more pragmatic approach to address the audio aspect of split rendering scenarios by defining three architectures.

## 3. Audio Architectures for Split Rendering Scenarios

It is assumed that there are two 5G entitites, one entity being the lightweight UE (AR glasses, XR device) and the other entity being the capable device (edge, smartphone). It’s also assumed that an XR scene comprises both visual and audio media. The visual media follows a split rendering approach, where decoding and (pre-)rendering are performed by a capable device (e.g., an edge server), and only lower-complexity processing is done on the lightweight UE. For the immersive audio media different constraints in terms of complexity and memory as well as constraints related to relevant interfaces between remote presentation engine and end device such as bit rate, latency, down- and upstream traffic characteristics may apply and therefore three different architectures for the audio component can be envisioned for split rendering scenarios:

### 3.1 Architecture 1: Local Audio Rendering

The immersive audio data is streamed directly to the lightweight UE, which is responsible for decoding, rendering, and synchronizing the audio with the post-rendered visual content. The lightweight UE processes the pose information locally and adjusts the audio rendering accordingly to create a convincing immersive experience. This approach provides low latency rendering and can preserve the high audio quality of the immersive media received by the Core Network but requires a higher computational capacity on the lightweight UE for decoding and rendering efficiently compressed immersive audio data. This also represents the case that should be considered as the reference.



Figure 1: Sequence of data flow for Architecture 1, Local Audio Rendering

### 3.2 Architecture 2: Distributed Audio Rendering

The capable device performs decoding and pre-rendering of the immersive audio media, and the pre-rendered audio is transmitted to the lightweight UE. The lightweight UE then performs decoding of the pre-rendered audio and post-rendering for pose correction, synchronizing the audio with the post-rendered visual content. The pose information is sent to the capable device, which adjusts the pre-rendering based on the pose data. This approach offloads computational load to the capable device from the lightweight UE and can offer low latency post-rendering. However, it comes at the expense of extra total complexity and requires a pre-rendering that fits to the traffic characteristics between the lightweight UE and the capable device to ensure the binaural audio corresponds to the latest pose after post-rendering.



Figure 2: Sequence of data flow for Architecture 2, Distributed Audio Rendering

### 3.3 Architecture 3: Remote Audio Rendering

The capable device is responsible for decoding and fully rendering the immersive audio media. The rendered audio is transmitted as coded binaural audio to the lightweight UE, which performs lightweight decoding without further pose correction of the rendered media. The lightweight UE synchronizes the binaural audio with the post-rendered visual content. This approach offloads most of the computational load to the capable device, resulting in the lowest downstream traffic and computational complexity on the lightweight UE. However, it may introduce higher motion-to-sound latency, depending on the coding used for the intermediate representation and the traffic characteristics, which could impact the immersive experience.



Figure 3: Sequence of data flow for Architecture 3, Remote Audio Rendering

## 4. Discussion

The source believes that those three architectures cover the split workflows mainly driven by the constraints of visual media.

The architectures differ in terms of complexity and memory, demands for the 5G link (including selection of appropriate 5QI), motion-to-sound and end-2-end latency, down- and upstream traffic characteristics, suitability for a given rendering scenario and ultimately in the solutions defined by the ISAR WI.

## 5. Conclusion

It is proposed to agree on the concept of three potential architectures for the audio media to address the need of the currently discussed split rendering workflows in other SA4 ongoing work. It is furthermore suggested to add section 3 to the draft TR on ISAR requirements.

## 6. References

[1] 3GPP S4-230434, WID on Immersive Audio for Split Rendering Scenarios