**3GPP TSG SA WG4 #127-bis-e meeting *-240580***

**Online, 08-14 April 2024**

|  |
| --- |
| *CR-Form-v12.2* |
| **pseudo CHANGE REQUEST** |
|  |
|  |  | **CR** |  | **rev** | - | **Current version:** | **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:***  | [SR\_MSE] pCR Adaptive Split Rendering Profile |
|  |  |
| ***Source to WG:*** | Nokia |
| ***Source to TSG:*** | S4 |
|  |  |
| ***Work item code:*** |  |  | ***Date:*** | 3rd Apr 2024 |
|  |  |  |  |  |
| ***Category:*** |  |  | ***Release:*** |  |
|  | *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-16 (Release 16)Rel-17 (Release 17)Rel-18 (Release 18)Rel-19 (Release 19)* |
|  |  |
| ***Reason for change:*** | Addition of advanced features to fulfill WID objectives. |
|  |  |
| ***Summary of change:*** | 1. Addition of a new Adaptive Split Rendering profile
 |
| ***tr*** |   |
| ***Consequences if not approved:*** | UE rendering resources will be under-utilized in split rendering and QoE may fluctuate substantially with operating conditions.. |
|  |  |
| ***Clauses affected:*** | Annex C.2 |
|  |  |
|  | **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:*** | N/A |

# Introduction and Motivation

Additional features for SR\_MSE have been presented to MBS multiple times over past meetings in [S4-231781](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/TSGS4_126_Chicago/Docs/S4-231781.zip), [S4aI230186](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/3GPP_SA4_AHOC_MTGs/SA4_MBS/Docs/S4aI230186.zip), [S4aI230206](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/3GPP_SA4_AHOC_MTGs/SA4_MBS/Docs/S4aI230206.zip), [S4aI230173](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/3GPP_SA4_AHOC_MTGs/SA4_MBS/Docs/S4aI230173.zip), [S4aI230207](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/3GPP_SA4_AHOC_MTGs/SA4_MBS/Docs/S4aI230207.zip), [S4-240187](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/TSGS4_127_Sophia-Antipolis/Docs/S4-240187.zip), [S4-240189](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/TSGS4_127_Sophia-Antipolis/Docs/S4-240189.zip), [S4-240195](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/TSGS4_127_Sophia-Antipolis/Docs/S4-240195.zip), [S4-240197](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/TSGS4_127_Sophia-Antipolis/Docs/S4-240197.zip) and most recently in [S4aI240011](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/Docs/S4aI240011.zip)**,** presented at SA4-e (AH) MBS SWG post 127 on 22nd Feb 2024. This contribution is a revision of [S4aI240011](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/Docs/S4aI240011.zip) based on previous discussions in the group, the comments received in the last meeting and the guidelines on adding new profiles in [S4aI240012](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/Docs/S4aI240012.zip). The main changes compared to [S4aI240011](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/Docs/S4aI240011.zip) are:

* Split-rendering session relocation has been removed
* The configuration and meta-data formats have been moved to the profile text
* The profile text is structured to better reflect the profile creation guidelines agreed to by the group.

While these features have been introduced and discussed in the past, including technical feasibility aspects (see [S4-240191](https://www.3gpp.org/ftp/TSG_SA/WG4_CODEC/TSGS4_127_Sophia-Antipolis/Docs/S4-240191.zip)), we revisit the main feature proposed in this document for the sake of completeness.

Split rendering, as implemented currently in TS 26.565, focusses on the simplest possible scenario where all the operations to render a scene graph are executed by the SRS, the rendered media is streamed to the SRC, which possibly applies pose correction to the rendered media and composes it into a display frame. This approach is useful for devices with limited rendering capabilities, but fails to leverage the considerable rendering capabilities of many device types in the market. Adaptive split rendering enables an SRC to render some objects of a scene graph locally in the device, while the rest of the objects are rendered at the SRS, while allowing the SRS and SRC to adjust the rendering split. The media rendered by the SRS is transmitted to the SRC which combines the rendered media from the SRS with locally rendered media to compose display frames. This approach may be used to improve end user QoE by:

* Leveraging rendering capabilities available at the device.
* Adjustment of the rendering split according to operating conditions.

These aspects are not just technically feasible, but well studied in academia, for example, [CloudVR](https://dl.acm.org/doi/10.1145/3240508.3240620), [Furion, Coterie](https://dl.acm.org/doi/abs/10.1145/3373376.3378516) and recently [RenderFusion](https://ieeexplore.ieee.org/abstract/document/10316520) (not to be confused with [Dual Render Fusion](https://docs.spaces.qualcomm.com/unity/setup/DualRenderFusionOverview.html) ) demonstrate feasibility as well advantages of this approach. Commercial remote rendering services like CloudXR and Azure Remote Rendering can both be configured to perform versions of adaptive split rendering. An opensource implementation using Unity game engine and WebRTC can be [found here](https://github.com/arenaxr/arena-renderfusion/tree/main), a video demo of the [implementation can be found here](https://www.youtube.com/watch?v=6mA4k9myuOM).

An important aspect that needs to be considered for adaptive split rendering is scene composition. There are different approaches for composing remotely rendered depending on the scene graph used, application logic and split management logic. In the simplest case, the remotely rendered frame can be used as a background for locally rendered objects. More complex split management and compositions are possible and we plan to bring contributions on generic implementation guidelines in the future.

Another aspect is state synchronization of objects in a 3D scene between the SRS and the SRC. For example, if a controller is being locally rendered by the SRC as in-game weapon and the user switches to a different weapon, this state change may need to be shared with the SRS to maintain a coherent application state.

We propose to incorporate the following text in TS 26.565.

1. Start of Change 1

# C.2 Adaptive Split Rendering Profile

## C.2.1 Introduction

This profile defines procedures and requirements for SRS and SRC to support split rendering features beyond a remote rendering paradigm.

Adaptive split rendering profile allows the SRC to render some objects of a scene locally and the rendering split can be adapted between the SRS and SRC during a session. The adaptation of the rendering split may be triggered either by the SRS or the SRC to maintain a consistent QoE of the SR session or to accommodate changes in operating conditions. The triggers may be, for example, channel conditions, SRC or SRS conditions or defined by the application provider.

To successfully render two parts of a scene separately in a split fashion, additional aspects of the rendering process need to be considered. Two basic requirements are maintaining a coherent state of the scene between the SRS and SRC and awareness of rendering split. Another requirement is seamless composition and display of the media rendered by the SRS and SRC into a frame to be displayed.

## C.2.2 Procedures and Call Flows

For adaptive split rendering, the general procedures and call flows in clause 5.2 are followed with the following additions and modifications.

* The SRS and SRC should share a scene description. The implementation details may vary. The application provider may decide whether to provide identical scene descriptions to the SRS and the SRC or whether to provide a truncated version of the scene description to the SRC.

Note: The Application Service Provider may provide the scene description resource to the SRS and SRC, for example, via RTC-8 to SRC and via RTC-2 to SRS.

* The SRS and SRC agree on an initial rendering split during session negotiation and the states to be synchronized, for example, in Step 5 of the procedure in clause 5.2.1.2.
* The initial rendering split and states to be synchronized are indicated in the SR configuration.
* In the rendering loop, exchange of split adaptation messages and state synchronization messages between the SRS and SRC is supported.

Figure C.2.2-1 illustrates a high level call flow set up and operation for a split rendering session which supports the adaptive split rendering profile.



 Figure C.2.2.‑1

The steps are :

1. In this optional step the SRC and the SRS acquire scene description of the scene to be rendered during the split rendering session. The actual implementation of delivery of the scene description by to the SRC and SRS is up to the application provider.
2. The Presentation Engine discovers the split rendering server and sets up a connection to it. It provides information about its rendering capabilities and the XR runtime configuration, e.g the OpenXR configuration may be used for this purpose. States to be synchronized and the initial rendering split is negotiated during this step.
3. In response, the split rendering server creates a description of the split rendering output and the input it expects to receive from the UE.
4. The Presentation Engine requests the buffer streams from the MAF, which in turn establishes a connection to the split rendering server to stream pose and retrieve split rendering buffers.
5. The Source Manager retrieves pose and user input from the XR runtime and state changes in negotiated states and possible requests from the Scene Manager.
6. a. The Source Manager shares the pose predictions and user input actions , state changes and possible split adaptation messages with the split rendering server.
7. a. The split rendering server uses that information to, update states, render the frame and possibly update the split.
8. b. The Scene Manager update states, renders a frame and possibly updates the split.
9. a. The rendered frame is encoded and streamed to the MAF.
10. b. Possible split adaptation and state change messages are shared with the presentation engine
11. The received media frames decoded and processed
12. The raw buffer frames are passed to the Scene Manager, this includes the frames received from the SRS and the frames rendered locally by the PE
13. The scene manager prepares composition layers and their corresponding swapchain images.
14. The swapchain images are forwarded to the XR runtime for composition and rendering
15. The frames are composed and displayed.

The final composition of a frame from media received from the SRS and locally rendered objects depends on the application logic. Implementation guidelines in C.2.7 provide a simple example.

## C.2.3 Metadata Formats

C.2.3.1 Split Rendering Configuration Format

The configuration format defined in clause 8.4.2.2 with the additional fields defined below shall be used for split rendering configuration exchange in adaptive split rendering profile.

|  |  |  |  |
| --- | --- | --- | --- |
| renderingSplit | Object | 1..1 | A object identifying objects to be rendered and where they are to be rendered (SRS or SRC), for example, as a dictionary with keys “SRS” and “SRC” and lists of object indices from a scene description or a scene graph  |
| synchronizedStatesInit | Object | 1..1 | A object identifying states to be synchronized between the SRS and SRC and their initial state |
|  states | Object  | 1..1 | A list of state identifiers, their current values |
|  state | String/number | 1..n | Identifier of a state |
|  initVal | String | 1..n | Initial value of the state |
|  stateVals | Array | 1..1 | An array of values possible for the state |

These renderingSplit object shall be present as part of the extraConfigurations Object as defined in clause 8.4.2.2 for extensibility of split rendering configuration format.

C.2.3.2 Split Adaptation Message Format

During a split rendering session, the operating environment of the split rendering server, the split rendering client or the network conditions may change. Consequently, the rendering split may need to be adapted to deliver a consistent QoE. When adaptive split rendering is enabled, the SRS or SRC shall request a new rendering split by sending a message of the type “**urn:3gpp:split-rendering:v1:asrp:sr-split”**. The message shall be conformant to the metadata message format specified in clause 8.5.1. The same message type shall be used to acknowledge, accept or reject the request by the receiver, with the message subtype identifying whether it is a request, acceptance, acknowledgement or rejection. The message shall follow the format in Table C.2.3.1-1.

Table C.2.3.2-1 Message format for split adaptation 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:split-rendering:v1:asrp:sr-split |
| message | Object | 1..1 | Message content  |
|  subtype | string | 1..1 | An identifier of the subtype of the message, it may be a request (REQ) for new split or acknowledgement (ACK), acceptance (OK) or rejection of a request (NOK). |
|  renderingSplitId | string | 1..1 | An identifier of the rendering split unique within the scope of the SR session |
|  renderingSplit | Object | 0..1 | A object identifying objects to be rendered and where they are to be rendered (SRS or SRC). The message shall be a dictionary object . with keys “SRS” and “SRC”, and values corresponding to a key shall be a list of named nodes from the scene description being rendered in the SR session. The keys shall indicate where the objects named in the corresponding value list are rendered.  |

Split adaptation messages indicating acceptance, acknowledgment or rejection of a split adaptation request may not include the renderingSplit Object.

C.2.3.3. State Synchronization Message Format

During a split rendering session, various states associated with the scene being rendered may transition. Depending on the nature of the application being executed, a transition may occur at the SRS, at the SRC or at both the SRS and SRC. For the application execution to be consistent, some state transitions need to be synchronized between the SRS and SRC. The SRC and SRS may agree on which states to synchronize during session setup. To synchronize state transitions during a split rendering session the SRS and SRC shall exchange messages of the type “**urn:3gpp:split-rendering:v1:asrp:sr-state”** . The same message type shall be used to send a state synchronization update, acknowledge a state synchronization update or simultaneously send and acknowledge a state synchronization update. The state synchronization update messages shall be conformant with the meta-data message format defined in clause 8.5.1 and the message content shall be formatted shown in Table C.2.3.2-1

Table C.2.3.2-1 Message format for state synchronization 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:split-rendering:v1:sr-state** |
| message | Object | 1..1 | Message content  |
|  subtype | string | 1..n | An identifier of the subtype of the message, it may be a state synchronization update (SYNC), acknowledgment (ACK) or both (SYNC\_ACK) |
|  syncUpdateId | string | 1..1 | An identifier of the synchronization update unique within the scope of the SR session |
|  synchronizedStates | Object | 1..1 | An object identifying states that are synchronized between the SRS and SRC and their current state. Only states that have transitioned may be exchanged |
|  states | Object  | 1..1 | A list of state identifiers, their current values and last change time |
|  identifier | String/number | 1..n | Identifier of a state |
|  val | Object/String/number | 1..n | Value of the state |
|  lastChangeTime | number | 1..1 | The timestamp of the last change in state |

Split adaptation messages indicating an acknowledgment of a state update may not include the synchronizedStates Object.

## C.2.4 SRC Capabilities

The adaptive split rendering profile may be used with pixel streaming profiles defined in Annex C.1. Profiles for two different scenarios are considered in Annex C.1: 2D pixel streaming for 2D or monoscopic use cases and 3D pixel streaming for stereoscopic use cases.

### C.2.4.1 Media Capabilities

If adaptive split rendering profile is used with the 2D pixel streaming profile, the SRC shall support the media capabilities of a device type 3 as defined in TS 26.119 [4], clause 10.4, and referenced in Annex C.1.2.2.2 .

If adaptive split rendering profile is used with the 3D pixel streaming profile, the SRC shall support the media capabilities for device type 1 as defined in TS 26.119 [4], clauses 10.2, and referenced in Annex C.1.3.2.2

If the device is a device type 2 as defined in TS 26.119 [4], clause 10.4, it shall also support the media capabilities of a device type 2 as defined in TS 26.119 [4], clause 10.3, and referenced in Annex C.1.3.2.2

If the device is a device type 4 as defined in TS 26.119 [4], clause 10.5, it shall also support the media capabilities of a device type 4 as defined in TS 26.119 [4], clause 10.5, and as referenced in Annex C.1.3.2.2

### C.2.4.2 Metadata Formats

**XR-Pose-Cap 1:** the SRC shall be able to retrieve one or more pose predictions for each view and for every frame to be rendered. The pose prediction shall be formatted according to clause 8.3.2.2.

**XR-Pose-Cap 2:** the SRC shall be able to retrieve and collect the user actions that occurred during an identified time interval. The action information shall be formatted according to clause 8.3.2.3.

**XR-ObjId-Cap 1**: the SRC shall be able to receive, retrieve and collect identifiers of objects in a scene being rendered by the SRC in a split rendering session during an identified time interval. The state information shall be formatted according to clause C.2.3.2

 **XR-ObjState-Cap 1:** the SRC shall be able to receive, retrieve and collect state changes in identified objects in a scene being rendered in an split rendering session during an identified time interval. The state information shall be formatted according to clause C.2.3.3

C.2.4.3 Rendering format description

The SRC and SRS shall comply with rendering format description in annex C.4.1

If adaptive split rendering profile is used for 3D pixel streaming, the SRC shall support the 3GPP\_node\_prerendered extension defined in C.4.2, However, the extension may be used on non-root nodes.

C.2.4.4 Scene Processing and Rendering CapabilitiesThe SRC shall have the following minimum scene processing capabilities:

* the **SD-Rendering-gltf-core** scene processing capabilities defined in clause 9.2 of TS 26.119.

**SD-Rendering-gltf-core** enables basic compatibility of an SRC with the adaptive split rendering profile for simple use cases, where the SRC does minimal local rendering and adaptability of rendering split is minimal. An example of such a limited scenario may be a cloud gaming use case where the application provider isolates a small subgraph of the complex game scene to be rendered by the SRC and shares the subgraph with the SRC. The subgraph may contain only the assets (mesh and textures) related to a user’s character and controller to allow the SRC to render these objects locally to mask motion to photon to render latency. More advanced use cases of adaptive split rendering place higher scene processing capabilities on the SRC. The SRC should have the following scene processing capabilities:

* the **SD-Rendering-gltf-Ext1** scene processing capabilities defined in clause 9.2 of TS 26.119.

In addition to the above specified scene processing capabilities, depending on the device type, the SRC shall have scene capabilities defined for each device type in clause 10 of TS 26.119

## C.2.5 SRS Capabilities

The SRS capabilities to support adaptive split rendering profile are described in the sub-clauses below.

### C.2.5.1 Media Capabilities

The media capabilities of the SRS are defined in relation to the media capabilities of the SRCs it is expected to serve. Therefore, the encoding capabilities of an SRS should match the decoding capabilities of the SRC.

If adaptive split rendering profile is used with 2D pixel streaming, for example, for monoscopic use cases, the SRS shall have capabilities defined in clause C.1.2.3

If adaptive split rendering profile is used with 3D pixel streaming, for example, for stereoscopic use cases, the SRS shall have capabilities defined in clause C.1.3.3.

### C.2.5.1 Metadata Capabilities

The SRS shall support the metadata formats for pose and action defined in Clause 8.3.2. In addition, the SRS shall support the metadata formats defined in Annex C.2.3, and complement the metadata capabilities defined in Annex C.2.4.2. This shall include the ability to receive and process messages corresponding to metadata capabilities defined in Annex C.2.4.2 and formatted according to clause 8.3.2 and C.2.3.

### C.2.5.1 Scene Processing and Rendering Capabilities

SRS shall have the **SD-Rendering-gltf-Ext1** scene processing capabilities.

Additionally, depending on the device type of the SRC participating in a split rendering session, the SRS should support the required and recommended scene processing capabilities defined in TS 26.119 in clause 10.3.5 for device type 2, clause 10.4.5 for device type 3, and 10.5.5 for device type 4.

## C.2.6 Profile identifiers

If the adaptive split rendering profile is used with monoscopic cases, for example with 2D pixel streaming profile defined in C.1.2, the type **urn:3gpp:sr-mse:src:profile:asr2dpixelstreaming** shall be included in splitRenderingProfile parameter when the SRC signals SRS the Split Rendering Configuration [8.4.2.2].

If the adaptive split rendering profile is used with stereoscopic cases, for example with 3D pixel streaming profile defined in C.1.3, the type **urn:3gpp:sr-mse:src:profile:asr3dpixelstreaming** shall be included in splitRenderingProfile parameter when the SRC signals SRS the Split Rendering Configuration [8.4.2.2].

## C.2.7 Extension to Client API Functions

TBA

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## C.2.7 Implementation Guidelines for Adaptive Split Rendering

TBA

1. End of Change 1