D1: Network Requirements for Metaverse Services

Network Requirements and Capabilities Working Group

Scope:

This deliverable provides a thorough analysis of relevant metaverse use-cases collected from diverse industry sources. It includes use-cases collection, mapping to processing scenarios, categorization and identification of technical and functional requirements.

The definition of architecture, interfaces or protocols is not in the scope of this document.

Versions:

Version	Authors	Note
1.0	Thibaud BIATEK (Nokia), Omar ELLOUMI (Nokia), Jens JOHANN (DT)	First version

Metaverse STANDARDS FORUM™

Table of Contents

References

[1] 3GPP SA4, "TS 26.81: Study on QoE metrics for AR/MR Services".

[2] 5GAA, "Tele Operated Driving (ToD): System Requirements Analysis and Architecture".

[3] M. Claypool and K. Claypool, "Latency and player actions in online games," 2006.

[4] PARSEC, "Nvidia NVENC Outperforms AMD VCE on H.264 Encoding Latency In Parsec Co-op Sessions," [Online]. Available: https://parsec.app/blog/nvidia-nvenc-outperforms-amd-vce-on-h-264-encoding-latency-in-parsec-co-op-sessions-713b9e1e048a.

[5] Meta Reality Labs, "Measuring the perception of latency with a haptic gloves," [Online]. Available: https://tech.facebook.com/reality-labs/2019/7/measuring-the-perception-of-latency-with-a-haptic-glove/.

[6] D. Wang, "Low latency object detection on the edge cloud," Uppsala Universiteit, 2021.

[7] Nokia, "Tackling the network challenges for XR: Understanding, developing and delivering an E2E XR connectivity solution," 2023.

[8] Blacknut, "Network connection configuration and optimisation," [Online]. Available: https://www.blacknut.com/en/support/network-requirements-optimization.

[9] Nvidia, "GeForce Now System requirements," [Online]. Available: https://www.nvidia.com/en-us/geforce-now/system-reqs/#:~:text=to%20properly%20function.-,Internet%20Requirements,we%20recommend%20less%20than%2040ms..

[10] 3GPP SA1, "TS 22.261: Service Requirements for the 5G System".

[11] A. Alnajim, S. Salehi, C.-C. Shen and M. Smith, "Traffic Characteristics of Extended Reality," 2023.

[12] M. Gapeyenko, V. Petrov, S. Paris, A. Marcano and K. I. Pedersen, "Standardization of Extended Reality (XR) over 5G and 5G-Advanced 3GPP New Radio".

[13] DAQRI, "Motion to Photon latency in mobile AR and VR," [Online]. Available: https://medium.com/@DAQRI/motion-to-photon-latency-in-mobile-ar-and-vr-99f82c480926.

Acronyms

AGV	Automated Guided Vehicle
AR	Augmented Reality
ATW	Asynchronous Time Warping
CDN	Content Delivery Network
FOV	Field of View
HD	High Definition
HMD	Head Mounted Display
MOS	Mean Opinion Score
QoE	Quality of Experience
QoS	Quality of Service
SD	Standard Definition
ToD	Tele-operated Driving
VR	Virtual Reality
XR	Extended Reality

1. Introduction

Metaverse services are wide in scope, spanning from industry and enterprise to consumer use-cases. The use-cases have different expectations, technically and functionally, but also involve different technologies, e.g. from extended reality to digital twinning. The involved devices can also be different, from acquisition, haptics gloves, headset, sensors, vehicles to smartphones. Most of the use-cases can be mapped to various processing scenarios, involving different computing and connectivity architecture.

To address this complex environment with an inclusive and technology-agnostic approach, it is important to separate each use-case from its possible architecture(s). This document addresses the use-cases analysis in a solution-agnostic manner, providing a set of generic processing scenarios on which each use-case can be mapped. This approach separates each use-case from specific network interfaces (e.g. 3GPP, WI-FI, etc ...) and focuses on functionality and pure connectivity requirements.

While use-cases can be diverse and address different needs, they will likely overlap in terms of connectivity requirements (e.g. delay, jitter and data rate) and functional needs (e.g. XR rendering, edge cloud). Thus, a categorization was done to consolidate similar use-cases and extract a reduced set of technical and functional requirements. To facilitate this consolidation work, a common and user-centric use-case template is used.

This deliverable is organized as follows. Section 2 collects use-cases, focusing on a user-experience centric approach. Section 3 provides background information on XR devices and quality of experience (QoE). Section 4 introduces processing scenarios, in a technology agnostic manner. Once the list of relevant use-cases is established, a consolidation and synthesis work will be conducted in Section 5, providing a concise list of functional and technical connectivity requirements. Section 6 concludes this work and establishes recommendations for the upcoming deliverables.

2. Use-cases

2.1. UC1: Maintenance support

Title
Maintenance support
Description
In today's industry, the technical complexity of workflows is increasing and although the degree of automation rises, human skills and dexterity are still needed for the delivery of a product and the deployment of a service. Especially in the area of telecommunications, outdoor activities of employees are still part of the daily job. Instead of looking up the necessary connections in a wiring diagram, the technician identifies the street cabinet (e.g. by a unique marker) and gets the information about the current pins to connect as seen from the viewpoint of his head-mounted display (see Figure 1).  *Figure 1: Wiring in a street cabinet*
Characterization of the use case from a user point of view:
Single user application, non-real time presentation necessary for diagrams, eventually streaming for videos Actor: • One technician, equipped with a head-mounted display System: • AR system with access to a server that contains information about the interior layout of a street cabinet, ticket database with open work items Interactions: • The technician uses the camera of the HMD to scan a tag that identifies the street cabinet in an unambiguous way.

- The server looks for open tickets for this street cabinet in the ticket database
- The server delivers to the AR system an overlay scheme of the pins of the street cabinet and marks those pins that need to be wired. The overlay can also include procedural 'How to' video clips to provide the technician with an overview of the expected operations to be performed.
- The AR system continuously maps the camera picture delivered by the HMD to the overlay scheme
- When the wiring is done, the technician marks this ticket as "done" on a virtual panel that is displayed in the HMD.

Characterization of the use case from a technical point of view:

Information gathering:

- Very likely all information delivered from one server, no necessity for edge computing, maybe interaction between physical and virtual objects, no distributed virtual environment.

Network aspects:

- Localized mobile service, unicast connectivity, low latency not necessary, no high data throughput, no handover into other networks, no need for strong synchronicity between different data streams. [should we also ask about a certain level of reliability or is this self-evident
- Latencies:
  - no critical values for static content and streaming of a "How to" video
  - Relocalization of overlay info in less than a second
- Data rates
  - uplink video of the HMD to the AR system, SD or HD resolution.
  - downlink rates depend on the format of the information and can range from a few hundred kbit/s for overlay graphics to a few Mbit/s for a video.

Target device(s):

- See-through HMD

Network exposure requirement(s)

- Quality on Demand to support carriage of media flows
- Network information / network insights to monitor network quality and react upon degradation
- Provisioning/deployment of computing resources

Reference to processing scenario(s):

Standalone rendering
Offloaded overlay generation

2.2. UC2: Avatar phoning

Title
Avatar phoning
Description
Two friends decide not to set up a video call but to be represented on the phone by an avatar. Each avatar is animated by the speech of the remote station as sketched in Figure 2.

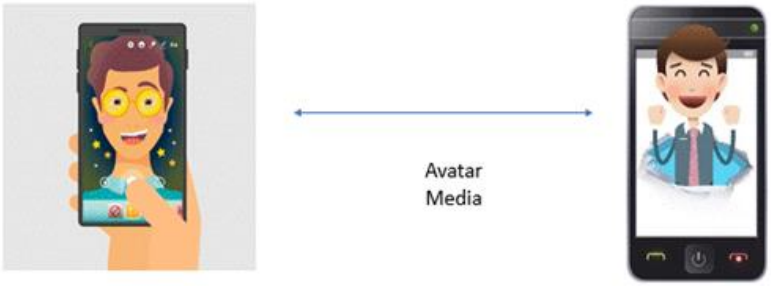


Figure 2: Avatar phoning

Characterization of the use case from a user point of view:
Single user application, real time presentation of the avatar Actors: • Two callers, equipped with a metaverse capable device System: • Telephone system, speech-operated character animation system, databank with (personalized) animation characters Interactions: • Before the call starts, the two callers select their representative animation character/avatar out of a databank • The speech of both callers is fed to a (network based) character animation system that analyzes the speech and creates phonemes out of it to animate a facial mask • The animated avatar is streamed from the character animation system to the remote phone • The speech can be synthesized to mimic the characteristics of the avatar or the original audio signal can be directly transferred to the remote phone. In the later case the original audio needs to be accurately synchronized with the animated mask.
Characterization of the use case from a technical point of view:
Information gathering:

• Only local information is used, i.e.: from device's sensors such as cameras and microphones
Network aspects:
• Unicast connectivity, low latency, low data throughput, handover into other networks possible, need for good synchronicity between animation and speech
• Latencies:
  ○ from caller to caller less than 100 ms
  ○ synchronicity between audio and animation
• Data rates:
  ○ Downlink/Uplink: audio and animated avatars
Target device(s):
• Smartphone
• VR/AR HMD
Network exposure requirement(s)
• Quality on Demand to support carriage of media flows
• Network information / network insights to monitor network quality and react upon degradation
• Network customization to ensure traffic prioritization for crucial data in case of congestion (e.g. audio over avatar data)

Reference to processing scenario(s):
Standalone rendering

2.3. UC3: Local interaction with offloaded processing

Title
Local interaction with offloaded processing
Description
A user is wearing a see-through XR headset and is navigating into the real-world where they are manipulating real or virtual objects. The headset is tracking its movements, and across sessions the headsets can be relocalized via anchors or via visual positioning in a (potentially globally aligned) shared map. The relocalization can happen on device or in the cloud. In the former case, relocalization information (map chunks) must be downloaded, and in the latter case, images and other sensor data must be sent to the cloud periodically. Virtual objects tied to the anchors, to the map, or to the geographic location are retrieved and shown to the user. This environment can be utilized for training (working environment, emergency situations, …) or remote support on the field, or in a fully automated environment including machine-learning based assistance and computer vision tasks. Typically, what the user can see is:

- real-time annotation on detected objects,
- real-time overlays on direction and positioning,
- rendered actions and visual effects telling him/her how to interact with the environment

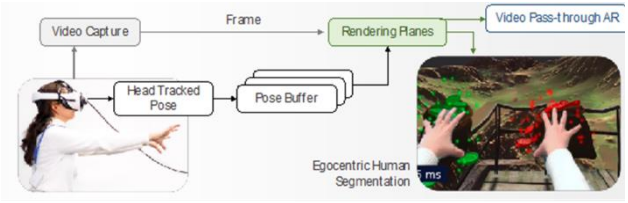


Figure 3: Local interaction assisted by offloaded processing

Characterization of the use case from a user point of view:

Single user application, real time interaction with physical and virtual objects, FOV is augmented with overlays and graphics coming from offloaded computer vision tasks (e.g. segmentation mask, object detection, ...).

Actors:

- Main user (on the field)

System:

- XR headset fully integrated, or tethered to a local device
- Communication network (wireless)
- Edge servers, to offload the computer vision tasks and host the immersive application with low latency processing

Interactions:

- Manipulation of physical or virtual elements
- Computer-vision generated overlays (segmentation maps, object tracking, ...)
- Overlayed and annotations placed by the remote user to help the main user

Characterization of the use case from a technical point of view:

Information gathering:

- Local information (camera, pose) is transmitted to the edge to run computer vision tasks
- Overlays generated by the computer vision tasks are transmitted to the main user

Network aspects:

- unicast connectivity, low latency, need for good synchronicity between animation and speech
- Latencies :
  - main-user -> edge cloud -> main user less than 100ms
- Data Rates:
  - Uplink : video and HMD sensors
  - Downlink: generated overlays

Target device(s):

- see-through HMD

Network exposure requirement(s)

- Quality on Demand to support carriage of media flows
- Network information / network insights to monitor network quality and react upon degradation
- Provisioning/deployment of computing resources at the edge – as close as possible
- Accurate location and precise positioning information, including indoor, to support real-time services.

Reference to processing scenario(s):

Offloaded overlay generation

2.4. UC4: Immersive telepresence

Title

Immersive telepresence

Description

A user is remotely attending a physical meeting. The user is alone wearing a headset. Other participants are together in the same meeting room in which a 360-degree camera is installed. The user may talk to other participants in this virtual reality space and the user's avatar is displayed in the meeting room. This can be achieved for training or an education, therapy or social/professional purpose.

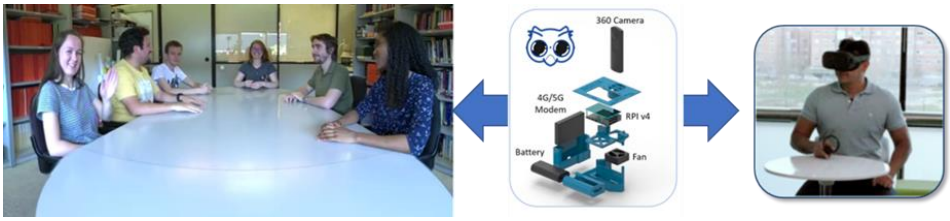


Figure 4: Immersive telepresence

Characterization of the use case from a user point of view:

Multi users application with remote user(s), real time A/V interaction, screen sharing, avatars, and connected metaverse users/endpoints.

Actors:

- Main users (in the meeting room)
- A remote participant

System:

- A device to display remote user avatar, including speakers (in the meeting room)
- A 360° acquisition device (in the meeting room)
- Communication networks in both the meeting room and the remote location
- XR headset fully integrated, or tethered to a local device (remote participant)
- Edge servers, to run the immersive application

Interactions:

- Speech with remote user
- Screen sharing
- Avatar sharing (including emotion)

Characterization of the use case from a technical point of view:

Information gathering:

- Local information (A/V, screen) is transmitted to the remote users
- Avatars and voice from remote users are transmitted to the main room

Network aspects:

- unicast connectivity, low latency, need for good synchronicity between animation and speech
- Latencies :
    - conference room -> remote user -> conference room less than 300ms
- Data rates:
    - Remote user
        - Uplink: audio and animated avatars
        - Downlink: 360 video and audio
    - Meeting room
        - Uplink: 360 video and audio
        - Downlink: audio and animated avatars

Target device(s):

- VR HMD

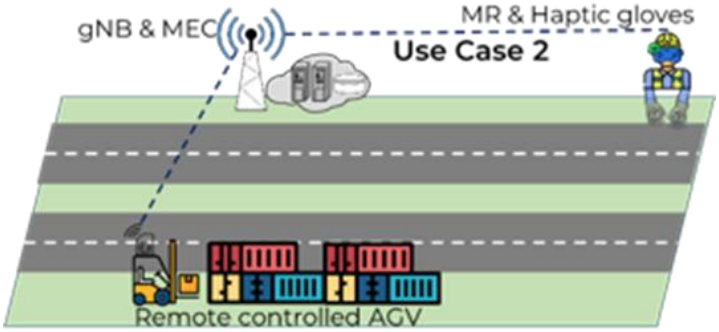
Network exposure requirement(s)

- Quality on Demand to support carriage of media flows
- Network information / network insights to monitor network quality and react upon degradation
- Network customization to ensure traffic prioritization for crucial data in case of congestion

Reference to processing scenario(s):

Split rendering
Standalone rendering

2.5. UC5: Immersive tele-operated driving

Title
Immersive tele-operated driving (ToD)
Description
The user is remotely operating a vehicle with an immersive cockpit displayed on VR head mounted display. The remote user assists autonomous vehicles that operate in hazardous environments from a remote, indoor and comfortable place, improving driver's safety. The user may manipulate a wide range of vehicles, from cranes to automobiles or robot arms. In order to properly feel its environment, the user is wearing haptic feedback devices. The user's FOV is enhanced with overlays to assist him during the operation, possibly incorporating annotated data coming from a digital twin of the remote location.    Figure 5: Immersive Tele-operated Driving (ToD)
Characterization of the use case from a user point of view:
Single user application, real time A/V interaction, haptic gloves/equipment such as force feedback steering wheel and pedals. The environment captured by the vehicle is enhanced with overlays and live information coming from a digital twin, and restituted to the remote driver. Actors: • Remote controlled AGV (on the field) • Remote driver System: • XR headset fully integrated with, or tethered to a local device • Haptics gloves • Communication network (wireless) • Edge servers, to run the application • Remote controlled AGV Interactions: • Haptics gloves

• Manipulation of physical of virtual elements • Driving
Characterization of the use case from a technical point of view:

Information gathering:
• Local information (A/V, screen, sensors) is transmitted to the remote user
• Driving commands are transmitted to the AGV
Network aspects:
• unicast connectivity, ultra low latency, high reliability
• Latencies :
  ○ 120ms roundtrip, 20ms ToD to HV and 100ms HV to HoD [1]
  ○ Note: depends on maximum vehicle speed, according to the 5G Automotive Association (5GAA) 200ms is required for low-speed maneuvers
• Data rates:
  ○ Average
  ▪ Uplink: audio, 360 degree video and sensors
  ▪ Downlink: vehicle controls
  ○ Remote driver:
  ▪ Uplink: vehicle controls
  ○ Downlink: audio, 360 degree video, haptics, overlays
Target device(s):
• VR HMD
Network exposure requirement(s)
• Quality on Demand to support carriage of media flows
• Network information / network insights to monitor network quality and react upon degradation
• Provisioning/deployment of computing resources at the edge – as close as possible
• Network customization to ensure traffic prioritization for crucial data in case of congestion
• Accurate location and precise positioning information, including indoor, to support real-time services.

Reference to processing scenario(s):
Split rendering for remote control with haptics

2.6. UC6: VR cloud gaming

Title

VR Cloud Gaming
Description
A user is playing a video game on its wireless XR device. The game can be either experienced in virtual or augmented reality. The user interacts with the game through some controllers and may wear haptic feedback devices.
Characterization of the use case from a user point of view:
Single user application, real time A/V interaction, haptic device. Actors: • Main user System: • XR headset fully integrated, or tethered to a local device • Controllers • Local communication network (wireless) • Edge servers, to run game engine Interactions: • Controllers • Head tracking, pose • Microphone/audio
Characterization of the use case from a technical point of view:

Information gathering:
• Local information (pose, control) is transmitted to edge cloud
• Rendered view and/or renderable data is streamed back to the main user
Network aspects:
• unicast connectivity, ultra low latency (game-type dependent)
• Latencies:
  ○ In the range of below 100ms to 1000ms end to end depending on the type of game [10]. First person games typically expect no more than 100ms while third person would require less than 1000ms.
• Data rates:
  ○ Uplink: pose and control information
  ○ Downlink: rendered video, audio and haptics
Target device(s):
• VR HMD
Network exposure requirement(s)
• Quality on Demand to support carriage of media flows

• Network information / network insights to monitor network quality and react upon degradation • Provisioning/deployment of applications at the edge
Reference to processing scenario(s):
Split rendering

2.7. UC7: Remote participation to a live entertainment event

Title
Remote and collaborative participation to a live entertainment event
Description
A venue (e.g. a stadium, a music festival) is equipped with an edge cloud based immersive acquisition system, enabling the event to be captured, produced and delivered, real-time, to a remote audience wearing XR headsets. A group of friends can join their dedicated virtual "VIP" box in the venue to attend the event. They can view each other's avatars and interact with each other while watching the event in an immersive manner. They may wear haptic devices to interact with the environment in a more realistic manner.
Characterization of the use case from a user point of view:
Multi-users application, real time A/V interaction, haptic gloves Actors: • Multiple users located across the globe System: • Many users, equipped with XR headset fully integrated, or tethered to a local device • Local communication network (wireless) Interactions: • Controller • Head tracking, pose
Characterization of the use case from a technical point of view:
Information gathering: • Camera feeds from the venue are transmitted to edge cloud to be turned into an immersive streaming service • Immersive streaming service is generated and distributed across a content distribution network

• User pose information, possibly controllers info are transmitted to the network
• Users communications are transmitted (including voice and avatars)
Network aspects:
• unicast connectivity, possibly multicast to optimize network, low-latency
• Latencies:
  ○ Similar case as first person interactive game (VR cloud gaming), i.e. less than 100ms end to end
• Data rates:
  ○ Uplink: pose, control information, animated avatars
  ○ Downlink: rendered video and audio, haptics
Target device(s)
• VR HMD
Network exposure requirement(s)
• Quality on Demand to support carriage of media flows
• Network information / network insights to monitor network quality and react upon degradation
• Network customization to ensure traffic prioritization for crucial data in case of congestion

Reference to processing scenario(s):
For the distribution part: • Split rendering • Standalone rendering

2.8. UC8: AR whiteboard

Title
AR Whiteboard
Description
Several users are working together on a virtual whiteboard. Each user is wearing a XR headset, the white board is placed in users' field of view (FOV). Users can draw or write on the whiteboard with their hands or with a connected pen. Users can talk to each other. Users can interact together as if they'd be in the same room in front of the white board. They can upload and manipulate content on the whiteboard (video, simulations, 3D models, images, ...).
Characterization of the use case from a user point of view:
Multiple users application, real time A/V interaction, haptic gloves

Actors:

- Users located across the globe

System:

- Many users, equipped with XR headset fully integrated, or tethered to a local device
- Local communication network (wireless)
- Edge servers, to run immersive application

Interactions:

- Controller, hands, gloves
- Head tracking, pose

Characterization of the use case from a technical point of view:

Information gathering:

- Avatars are transmitted
- Whiteboard information and interactions are transmitted
- Speech communications are achieved
- User pose information, possibly controllers info are transmitted

Network aspects:

- unicast connectivity, low-latency, synchronization of avatars/whiteboard/users/voice
- Latencies:
    - Similar case as first person interactive game, i.e. less than 100ms end to end
- Data rates:
    - Uplink: pose, control information, assets, animated avatars
    - Downlink: Rendered audio and video

Target device(s):

- See-through HMD

Network exposure requirement(s)

- Quality on Demand to support carriage of media flows
- Network information / network insights to monitor network quality and react upon degradation
- Provisioning/deployment of computing resources at the edge – as close as possible

Reference to processing scenario(s):

Split rendering for collaborative work

3. Background

3.1. XR device

Before describing in detail the possible processing scenarios, this section documents what is an XR-capable device. This description relies on the work conducted in 3GPP-SA4, particularly in TS 26.119 [1]. It is assumed that our XR devices are composed by several internal components:

- The **XR Application** is a software responsible for proper integration of virtual signals (audio, video, haptics, overlays, ...) into a user's real-world environment.
- The **XR Runtime** is a set of functions providing to the XR application access to controller and peripherals, whether it is for acquisition, probing or display.
- The **Device and Media APIs** provide access to media encoders, decoders, packaging, encryption, synchronization, rendering, etc.
- The **Network connectivity** providing access to multiple possible access networks including, but not limiting to, 5G, WiFi, Broadband.

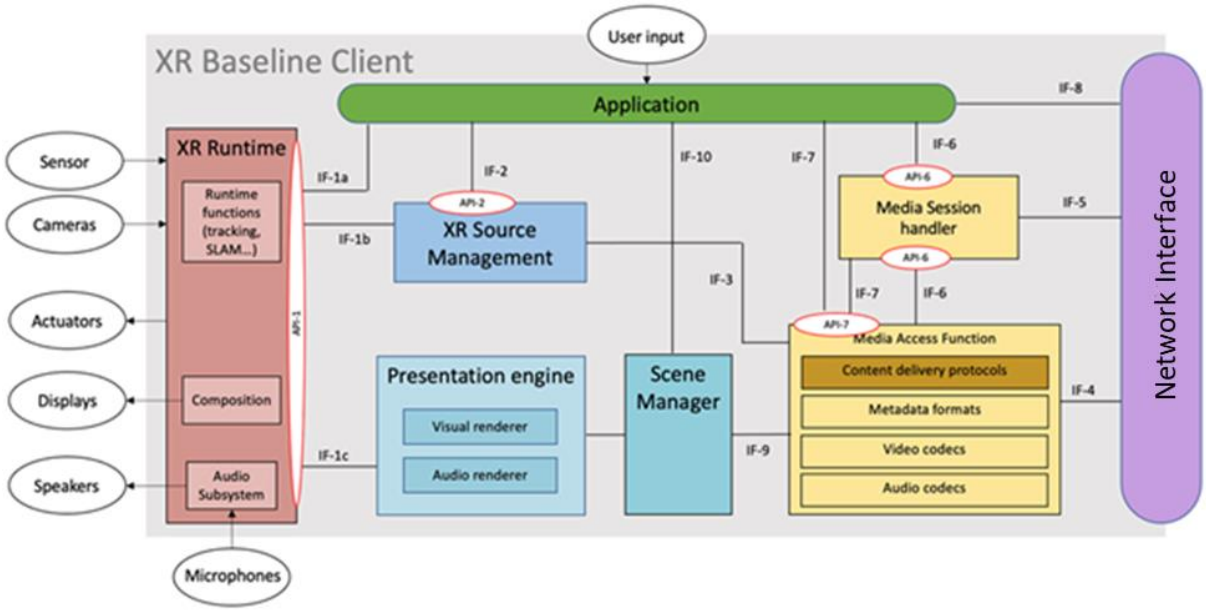


Figure 6: Components of the XR-capable device, as defined by 3GPP TS 26.119 [1]

3.2. QoE considerations

The QoE has been thoroughly studied in the literature, for a wide range of applications, from IPTV to streaming or conversational services. The ITU has defined in [2] the QoE as "The overall acceptability of an application or service, as perceived subjectively by the end-user". The dimensions of QoE documented by ITU in this report is depicted in the Figure 7, and is composed of objective and subjective elements. The commonly used metric to measure the QoE of an audiovisual service is the Mean Opinion Score (MOS) and can be captured via subjective testing using standardized methodologies (e.g. in [3] [4]).

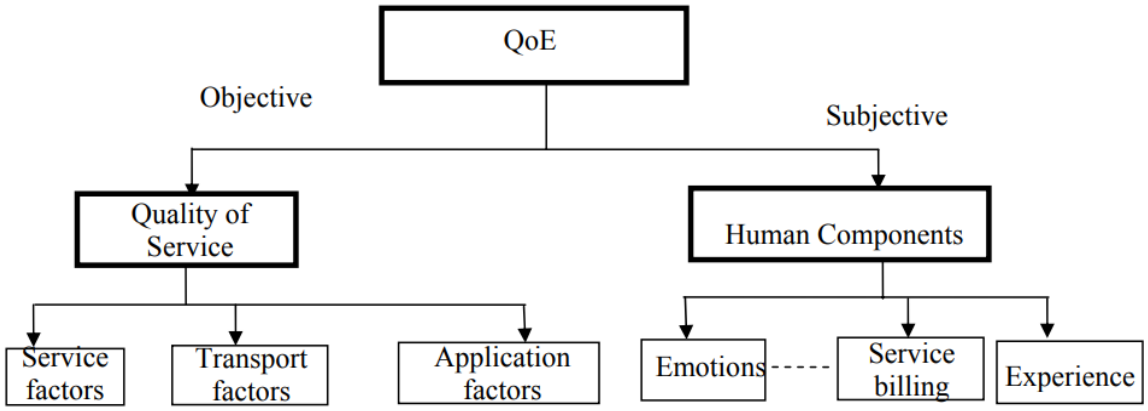


Figure 7: QoE components, as defined by ITU-T G.1080 [2]

The 3GPP SA4 has worked on the QoE as well, in TR 26.909 [5] and TR 26.928 [6], respectively for TV services and XR. From [5], a list of typical streaming metrics is identified as sufficient KPIs to characterize dropping video streaming QoE:

- stalling: initial, but also periodic and freezing while playing,
- interruption of audio while playing,
- visible video artifacts, including blurring, blocking and mosquito artifacts,
- varying video quality while playing.

While this can be sufficient to characterize QoE of traditional video streaming services, [7] has documented additional components pertaining to XR-based services, associated with the "immersiveness and presence". In terms of technical metrics, this translated into visual, auditory, and sensor/haptic quality parameters.

The visual quality can be measured with the following metrics:

- tracking, in terms of degrees of freedom (DoF), accuracy (centimeter, degrees), jitter, frequency.
- latency, in terms of motion-to-photon, pose-to-render, processing loop and interaction delays.
- persistence, in terms of pixel and display refresh rate.
- spatial and temporal resolution, in terms of pixels per inches (PPI) and framerate.
- optics, in terms of FoV (degrees

The auditory quality can be measured as follows [7]:

- latency, in terms of motion-to-sound latency
- binaural sound reproduction with undistorted magnitude and phase frequency responses
- stable and accurate sound localization properties in an omnidirectional space
- transparent or near-transparent audio quality

The sensor/haptics quality can be measured as follows [8]:

- latency, in terms of "touch-to-feel"
- tracking, in terms of degrees of freedom (DoF), accuracy (centimeter, degrees), jitter, frequency.
- resistance and pressure

It is expected that some use-cases may require "presence". The required level of presence will set the cursor on the quality indicators described above. For instance, the cloud VR gaming scenario

requires a sensation of presence to be immersed into the virtual environment while the uses-cases on overlay generation would not.

Recently, 3GPP SA4 initiated a new study item on QoE metrics for AR/MR services [9] for Rel. 18. In this technical report, SA4 defines observation points within the XR device documented in the previous section, and illustrated in Figure 7.2.2. These observation points (labeled as OP-1, OP-2, OP-3 and OP-4) are important to better define and understand the network latency requirements that will be specified for our use-cases.

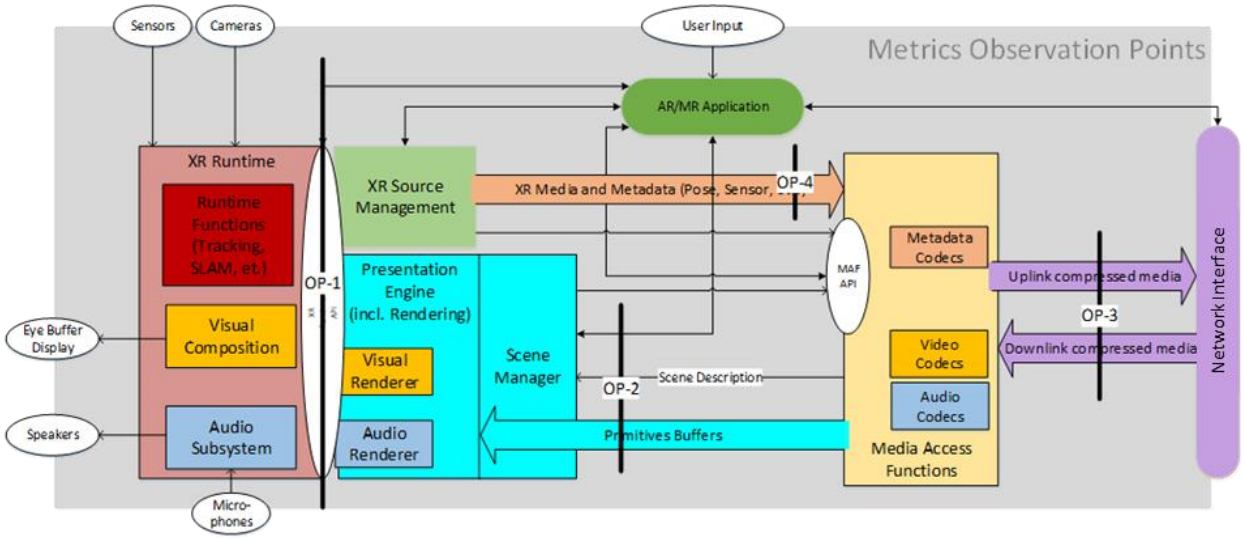


Figure 8: AR/MR metrics observation points, as defined by 3GPP TR 26.812 [9]

4. Processing scenarios

4.1. Preamble

It is expected that the required processing for the abovementioned use-cases might be hosted on the device, or partially or even fully shifted to a remote location. This deployment choice is driven by the device's available power and targeted use-cases. The figure below illustrates the different envisaged processing modalities from local processing to cloud and split-rendering, where the application, XR runtime, scene manager or media access function can be more or less offloaded to a remote location.

Figure 9: Evolution of processing modes [10]

As a given use-case can be deployed in many different processing ways, this section provides a collection of processing scenarios on which the use-cases will be mapped to. A functional approach has been taken in 3GPP SA4 [6] [1] to define generic architecture, focused on the XR-related aspects. The processing scenario is mainly driven by the end-device capability in terms of computational power, required to run rendering operation or computer vision tasks. The following subsections provide considerations on XR device and QoE aspects followed by processing scenarios and establish to which of the abovementioned use-cases they are mapped.

4.2. Scenario 1: Split rendering

Raster-based split rendering refers to the case where the XR Server runs an XR engine to generate the XR Scene based on information coming from an XR device. The XR Server rasterizes the XR viewport and does XR pre-rendering. It applies to the following use-cases:

- VR Cloud Gaming
- Remote and collaborative participation to a live entertainment event
- Immersive telepresence

According to Figure 10, the viewport is predominantly rendered in the XR server, but the device is able to do latest pose corrections, for example by asynchronous time-warping or other XR pose correction to address changes in the pose.

XR graphics workload is split into rendering workload on a powerful XR server (in the cloud or the edge) and pose correction (such as ATW) on the XR device.

Low motion-to-photon latency is preserved via on-device Asynchronous Time Warping (ATW) or other pose correction methods.

As ATW is applied the motion-to-photon latency requirements (of at most 20 ms) can be met by XR device internal processing. What determines the network requirements for split rendering is time of pose-to-render-to-photon and the roundtrip interaction delay. This determines the latency requirements for the 5G delivery.

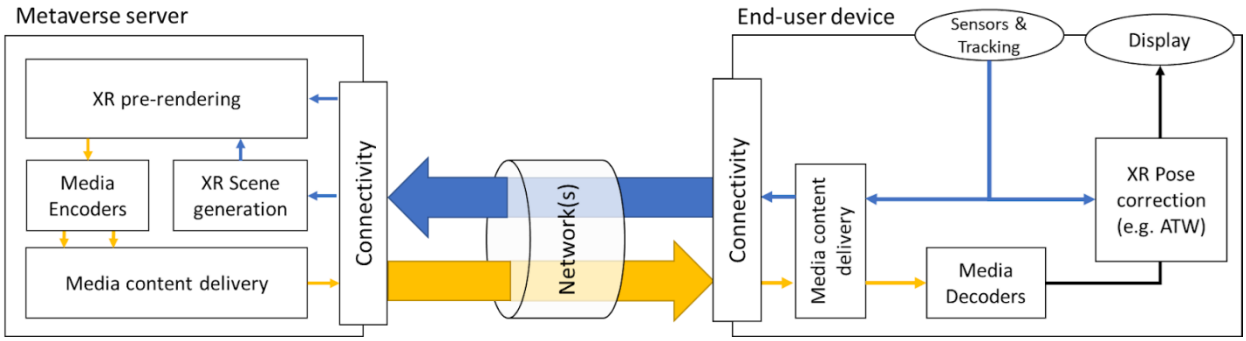


Figure 10: Split Rendering with pose correction

The following call flow highlights the key steps:

1. An XR device connects to the network and joins XR application
    i. Sends static device information and capabilities (supported decoders, viewport)
2. Based on this information, the XR server sets up encoders and formats
3. Loop
    i. XR device collects XR pose (or a predicted XR pose)
    ii. XR Pose is sent to XR server
    iii. The XR server uses the pose to pre-render the XR viewport
    iv. XR viewport is encoded with 2D media encoders
    v. The compressed media is sent to XR device along with XR pose that it was rendered for
    vi. The XR device decompresses video
    vii. The XR device uses the XR pose provided with the video frame and the actual XR pose for an improved prediction and to correct the local pose, e.g. using ATW.

In terms of formats and protocols, a real-time delivery protocol is used to deliver the images uplink and downlink (e.g. RTP-based, WebRTC). A low-latency image/video codec is used to compress the video signals (e.g. JPEG XS, or a traditional codec HEVC/VVC with low-latency GoP).

Finally, the relevant QoE parameters are:

- Viewports stability responsiveness to user's head movements. This is captured by:
    - tracking, in terms of degrees of freedom (DoF), accuracy (centimeter, degrees), jitter, frequency.
    - latency, in terms of motion-to-photon, pose-to-render, processing loop and interaction delays.
- Viewports quality. This is captured by:
    - spatial and temporal definition, in terms of resolution and framerate.
    - persistence, in terms of pixel and display refresh rate.
    - optics, in terms of FoV (degrees) and display in terms of pixels per inches (PPI)

4.3. Scenario 2: Offloaded overlay generation

Alternatively to scenario 1, this scenario handles cases where the see-through viewer field of view is enhanced based on side-processing. The views are not rendered in the server, but the overlay to be put on top of the view is offloaded as it may involve computational-intensive algorithms. In this scenario, the end-user device sends to the server video acquisition (`what I see`). This data is then processed with complex models which generate an overlay (including animated elements). This is streamed down to the device which incorporates it in the user's field of view. It applies to the following use-cases:

- Maintenance support.
- Local interaction with offloaded processing.

Figure 11 depicts baseline architecture for offloaded overlay generation.

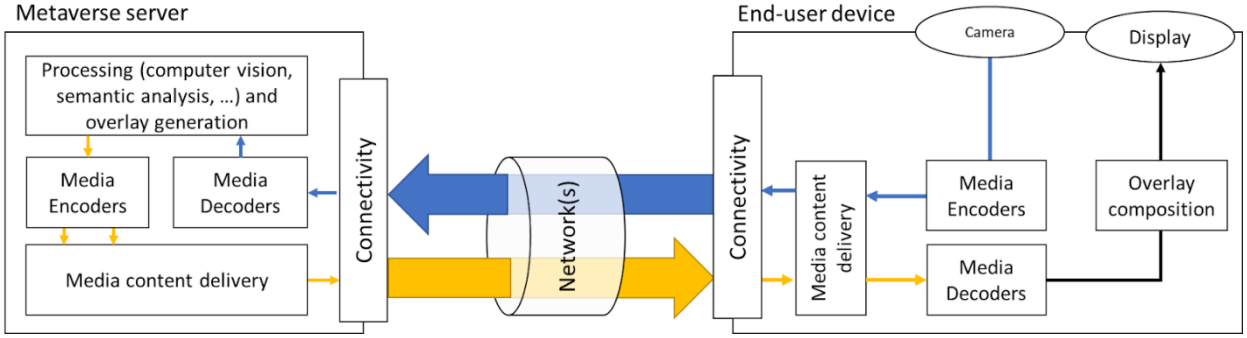


Figure 11: Offloaded overlay generation

The following call-flow highlights the key steps:

1. An XR device connects to the network and joins XR application
    i. Sends static device information and capabilities (supported decoders, viewport)
    ii. Sends static information about required processing to be run and the desired input/output
2. Based on this information, the XR server sets up processing, encoders and formats
3. Loop
    i. XR device acquires image through the camera
    ii. XR device encodes the image
    iii. XR device sends compressed image to the server
    iv. The XR server decodes the image
    v. The XR server runs required processing and produces required output
    vi. The XR server encodes the output overlay(s)
    vii. The compressed overlay(s) is sent back to the XR device along with any side metadata useful for the application
    viii. The XR device decompresses the overlay(s)
    ix. The XR device maps the overlay(s) to its FoV

In terms of formats and protocols, a real-time delivery protocol is used to deliver the images uplink and downlink (e.g. RTP-based, WebRTC). A low-latency image/video codec is used to compress the video signals (e.g. JPEG XS, or a traditional codec HEVC/VVC with low-latency GoP).

Relevant QoE parameters are:

- Alignment of the generated overlays with the user's FoV. This is captured by the following parameters:
  - tracking, in terms of degrees of freedom (DoF), accuracy (centimeter, degrees), jitter, frequency.
  - latency, in terms of motion-to-photon, pose-to-render, processing loop and interaction delays.
- Overlay quality. This is captured by the following parameters:
  - spatial and temporal definition, in terms of resolution and framerate.
  - persistence, in terms of pixel and display refresh rate.
  - optics in terms of FoV (degrees) and display in terms of pixels per inches (PPI)

4.4. Scenario 3: Standalone rendering

Some devices are expected to be capable enough to render and execute all processing tasks locally, without need for a bidirectional communication. The connectivity is used to download assets and models used for local rendering and processing. It applies to the following use-cases:

- Maintenance support
- Remote and collaborative participation to a live entertainment event
- Immersive telepresence

Figure 12 depicts baseline architecture for standalone rendering.

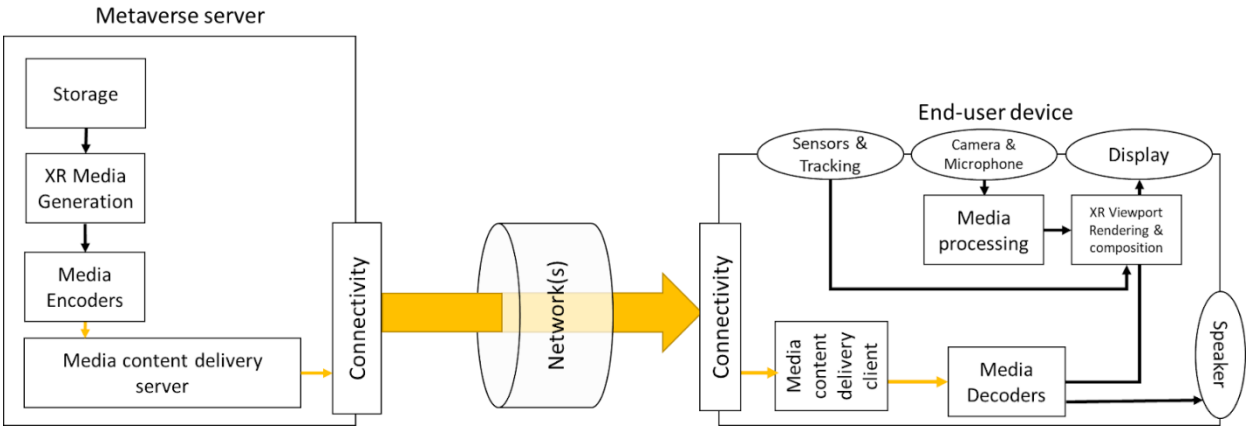


Figure 12: Standalone rendering

The following call-flow highlights the key steps:

1. The XR device configure its initial parameters (e.g. in case of avatar communications, a pre-scan of the user face can be achieved)
2. XR device connects to the network and joins the XR application
   i. Sends static device information and capabilities (supported decoders, viewport)
   ii. Downloads required assets for the application (e.g 3D models)
3. Based on this information, the XR server sets up the shared 3D asset, the encoders and formats
4. Loop
   i. The XR device acquires pose

ii. If needed, additional interfaces are captured (e.g. video) to be processed in order to generate overlay
iii. The XR pose and, if needed the overlays, are used to render the XR viewports
iv. If needed, new assets are downloaded (e.g. 3D models)

Depending on the use-case in context, various contents formats and delivery protocols can be used. For example, a non-real time download would not involve the same formats and protocols as a live 3D experience streamed from a venue.

Relevant QoE parameters are:

- As the viewports generation is done on the device, the stability and responsiveness of user's head movement should be ensured and is not impacted by connectivity.
- Viewports quality. This is captured by the following parameters:
  - spatial and temporal resolution of the delivered 3D assets, in terms of pixels per inches (PPI) and framerate.
  - persistence, in terms of pixel and display refresh rate.
  - optics, in terms of FoV (degrees)

4.5. Scenario 4: Split rendering for collaborative work

This processing scenario is for use-cases where inputs from multiple viewers impact the XR scene which is rendered. It applies to the AR whiteboard use-case of this deliverable.

Figure 13 depicts baseline architecture for combining split rendering for collaborative work.

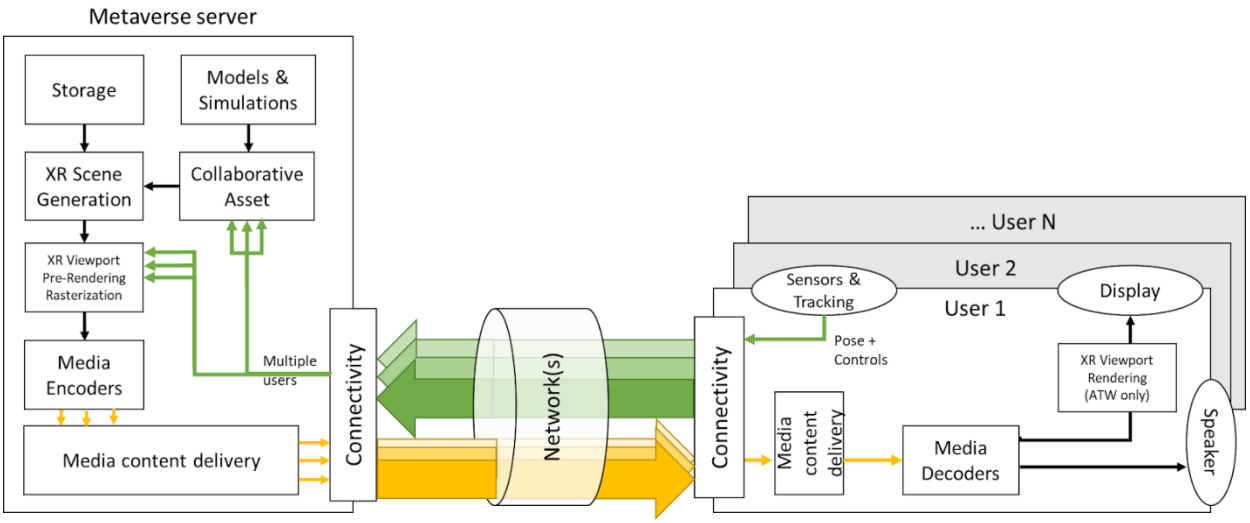


Figure 13: Split rendering for collaborative work

The following call-flow highlights the key steps:

1. Multiple XR devices connect to the network and join the XR application
   i. They send static device information and capabilities (supported decoders, viewport)
2. Based on this information, the XR server sets up the shared 3D asset, the encoders and formats
3. Loop

i. XR Device collects XR pose (or a predicted XR pose) and interaction with the 3D asset
ii. XR Pose and interactions are sent to the XR Server
iii. The XR server uses the interaction parameters to update the 3D asset accordingly
iv. The XR Server uses the pose to pre-render the XR viewport based on the updated 3D asset
v. XR Viewport is encoded with 2D media encoders
vi. The compressed media is sent to XR device along with XR pose that it was rendered for
vii. The XR device decompresses video
viii. The XR device uses the XR pose provided with the video frame and the actual XR pose for an improved prediction and to correct the local pose, e.g. using ATW.

In terms of formats and protocols, a real-time delivery protocol is used to deliver the rendered views downlink (e.g. RTP-based, WebRTC). An efficient video codec should be used to increase bandwidth efficiency (e.g. HEVC or VVC) while providing low-latency coding structures. Interaction with the 3D assets may involve motion, pictures, video, equations, models, etc … more investigations required.

Relevant QoE parameters are:

- Interaction between users on the model should be seamless. This is captured by the following parameters:
  - latency, in terms of motion-to-photon, pose-to-render, processing loop and interaction delays.
  - uplink capacity, to absorb highly demanding data uploads.
- Viewports stability responsiveness to user's head movements. This is captured by:
  - tracking, in terms of degrees of freedom (DoF), accuracy (centimeter, degrees), jitter, frequency.
  - latency, in terms of motion-to-photon, pose-to-render, processing loop and interaction delays.
- Viewports quality. This is captured by:
  - spatial and temporal definition, in terms of resolution and framerate.
  - persistence, in terms of pixel and display refresh rate.
  - optics, in terms of FoV (degrees) and display in terms of pixels per inches (PPI)

4.6. Scenario 5: Split rendering for remote control with haptics

This processing scenario is for use-cases involving remote control of vehicles or robots, the metaverse server is managing communications between several entities having different QoE and QoS requirements. In addition to the rendering, haptics and controls communications are managed. It applies to immersive ToD in this deliverable.

Figure 14 depicts baseline architecture for combining split rendering with remote control and haptics.

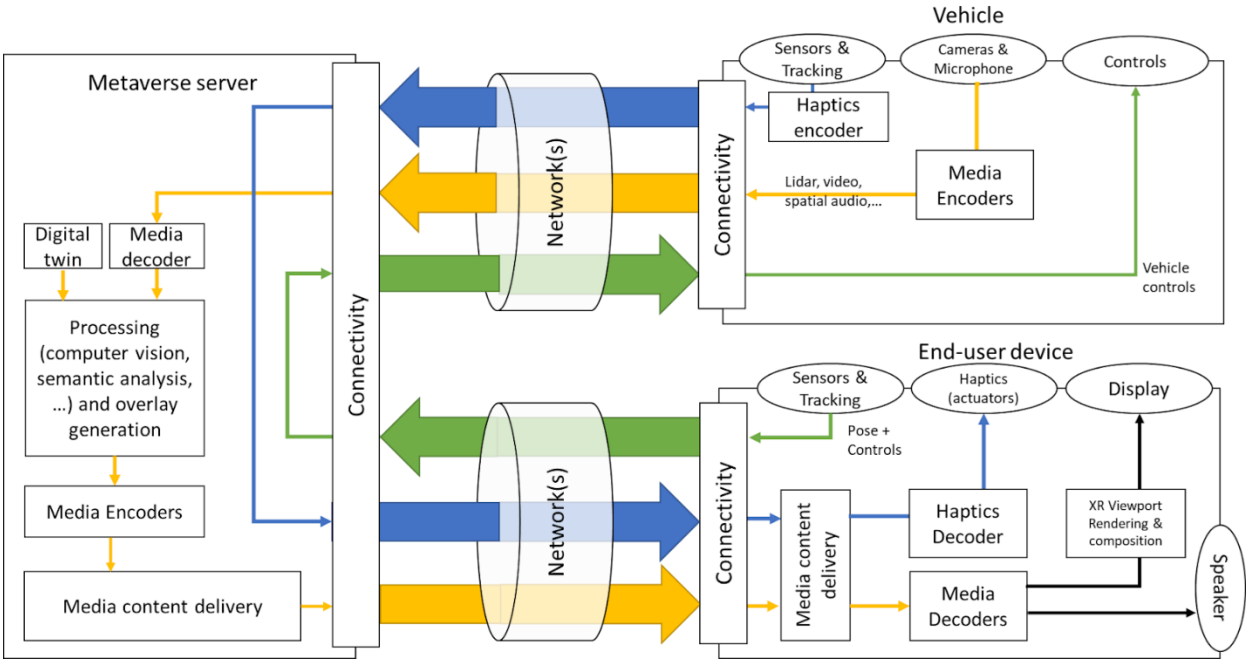


Figure 14: Split rendering for remote control with haptics

The following call-flow highlights the key steps:

1. A XR device connect to the network and join the XR remote driving application
    i. Sends static device information and capabilities (supported decoders, viewport)
    ii. Sends static information about required processing to be run and the desired input/output
2. The remote vehicle connect to the network and join the XR remote driving application
    i. Sends static device information and capabilities (supported video encoders, haptics encoders)
3. Based on this information, the XR server sets up processing, encoders/decoders, formats and connects to relevant digital twins of the remote environment.
4. Loop
    i. The vehicle acquires spatial video & audio, haptics, position, and send it to the server
    ii. The data flows are encoded with low-latency codecs
    iii. The server is passing through video and audio to the XR device but is also running processing tasks to generate overlays that sent to the user
    iv. The XR device receives all the data flows
    v. The XR device decodes and renders haptics
    vi. The XR device decodes and renders spatialized audio
    vii. The XR device decodes video & overlay, composes the final XR and renders the views.
    viii. The XR device acquires vehicle control information triggered by the user.
    ix. The control information is sent to the vehicle and is passing to the vehicle control system

In terms of formats and protocols, a real-time delivery protocol is used to deliver the images uplink and downlink (e.g. RTP-based, WebRTC). A low-latency image/video codec is used to compress the video signals (e.g. JPEG XS).

Relevant QoE parameters are:

- Alignment of the generated overlays with the user's FoV. This is captured by the following parameters:
  - tracking, in terms of degrees of freedom (DoF), accuracy (centimeter, degrees), jitter, frequency.
  - latency, in terms of motion-to-photon, pose-to-render, processing loop and interaction delays.
- Alignment of a digital twin with real world video acquisition. This is captured by the following parameters:
  - positioning, in terms of accuracy (centimeter, degrees)
- Remote vehicle control responsiveness. This is capture by the following parameters:
  - A/V latency, in terms of motion to photon
  - Haptics feeling, in terms of actuators accuracy (pression)
- Overlay quality. This is captured by the following parameters:
  - spatial and temporal definition, in terms of resolution and framerate.
  - persistence, in terms of pixel and display refresh rate.
  - optics, in terms of FoV (degrees) and display in terms of pixels per inches (PPI)

5. Use-cases classification and requirements

In this section, a classification of the use-cases described in Section 6 is proposed. Instead of dealing separately with each use-case, the use-cases are classified based on the processing scenarios documented in Section 4. The QoE parameters, translated into technical network requirements, are consolidated. The network requirements are refined based on targeted XR device architecture provided in Section 3.1 and considering the observation points and metrics defined in Section 3.2.

5.1. Mapping of use-cases to processing scenarios

Eight use-cases were collected in Section 2, some of them having features in common. To enable these use-cases to be deployed, different possible processing scenarios are defined in Section 4. The processing scenarios provide potential function-based solutions for a use-case to be implemented, defining reference architecture at a high level. The eight use-cases were mapped to five processing scenarios, as highlighted in Figure 9.1-1.

	Split rendering	Offloaded Overlay Generation	Standalone Rendering	Split-rendering for collaborative work	Split-rendering for remote control with haptics
Maintenance support		X	X		
Avatar phoning			X		
Local interaction with offloaded processing		X			
Immersive telepresence			X	X	
Immersive Tele-operated Driving (ToD)					X
VR Cloud gaming	X				
Remote and collaborative participation to a live entertainment event	X		X		
AR Whiteboard				X	

Figure 15: Mapping between use-cases and processing scenarios

5.2. Mapping of QoE expectations to requirements and KPIs

5.2.1. Split-rendering

Two use-cases can be deployed using a split-rendering mechanism:

- UC-6: VR cloud gaming
- UC-7: Remote and collaborative participation to a live entertainment event

Those two use-cases require feeling of presence with their environment, but also with other participants. The main drivers for enabling presence for these use-cases are related to system responsiveness (motion to photon) and quality of rendered scenes. As documented in Section 4.2, this requires an accurate tracking, an excellent PPI density, a wide FoV angle and a high-quality rendering. From a network perspective, this translates from throughput and latency metrics. Latency takes into account internal XR device architecture that can introduce their own delay budget to the motion to photon criterion.

Table 1: Network requirements for split-rendering processing scenario

	Session aspects			Throughput					Latency (worst-case)
	#Users	Device Speed	Distance between users	Video	Audio	Sensors	Haptics	Data	
UC6	[1-100]	Static	Worldwide	Rendering: [6-35]Mbps DL [10] [11]	500kbps DL	Pose 500Kbps UL	500kbps DL	Controls: 500kbps UL	"Worst-case" is first person game [12] + [60-100]ms motion-to-render-to-photon [Device] -1ms for sensors and pose acquisition [13] [Server] -(1/Fr)ms for scene generation and rendering (Note: depends on rendering targeted framerate, 16ms for 60fps) [Server] -6ms for GPU encoding [14] [Device] -20ms for decoding (1 frame at 50fps) [Device] -ems for composition and display =~[17-57]ms remaining for the network
UC7	[2-10]	Static	Worldwide	Rendering: 20Mbps DL	500kbps UL 500kbps DL	Pose 500Kbps UL	500kbps DL	Controls: 500kbps UL	

In addition to these technical aspects, additional functional requirements shall be met by the connectivity standards in order to ensure an excellent application level QoE:

- Flexible throughput management, to adapt bitrate to congestion
- Capability to prioritize sub-flows compared to other ones, based on their relative importance

5.2.2. Offloaded overlay generation

Two use-cases can be deployed using offloaded overlay generation scenario:

- UC1: Maintenance support
- UC3: Local interaction with offloaded processing

These two use-cases do not require presence. The main drivers to reach expected QoE, as documented in clause 4.3 are the capability of the system to provide accurate enough tracking as the overlay needs to be accurately positioned, and to deliver a high-quality overlay. While the tracking accuracy is left to the device, the connectivity shall provide low enough latency to minimize QoE drops that can be introduced by pose correction. Furthermore, a sufficient shall be supported, to enable uplink transmission of acquired video and downlink delivery of high-quality overlays.

Table 2: Network requirements for offloaded overlay generation scenario

	Session aspects			Throughput					Latency (worst-case)
	#Users	Device Speed	Distance between users	Video	Audio	Sensors	Haptics	Data	
UC1	[1]	Static	N/A	Overlay: [0.5-10] Mbps DL Video: <= 3Mbps UL	N/A	Pose 500Kbps UL	N/A	Instruction: 3Mbps DL	1000ms for overlay placement [Device] -1ms for sensors and pose acquisition [13] [Device] -20ms for acquisition and video encoding (1 frame at 50fps, can be lower with ultra low latency/high throughput codecs such as JPEG XS) [Server] -40ms for computer vision tasks [Server] -6ms for GPU encoding [14] [Device] -20ms for decoding (1 frame at 50fps) [Device] -ɛms for composition and display = ~913ms remaining for the network
UC3	[1]	Static	N/A	Overlay: [1-10] Mbps DL Video: [5-20] Mbps UL	500kbps UL 500kbps DL	Pose 500Kbps UL	N/A	N/A	"Worst-case" is live overlay generation + 100ms motion-to-render-to-photon [Device] -1ms for sensors and pose acquisition [Device] -20ms for acquisition and video encoding (1 frame at 50fps) [Server] -40ms for computer vision tasks [Server] -6ms for GPU encoding [14] [Device] -20ms for decoding (1 frame at 50fps) [Device] -ɛms for composition and display = ~13ms remaining for the network

5.2.3. Standalone Rendering

Three use-cases can be deployed using a standalone rendering scenario:

- UC1: Maintenance support
- UC2: Avatar phoning
- UC4: Immersive telepresence

For this processing scenario, the device is almost the only responsible for ensuring high QoE. Connectivity in this scenario will only be used to carry out 3D assets used for the rendering and processing inside the devices. Hence, the network is mainly expected to be able to download assets.

Table 3: Network requirements for standalone rendering scenario

Session aspects	Throughput	Latency (worst-case)

	#Users	Device Speed	Distance between users	Video	Audio	Sensors	Haptics	Data	
UC1	[1]	Static	N/A	Picture [0.5-3]Mbps UL Overlay asset: [1-5]Mbps DL	N/A	N/A	N/A	Instruction: 3Mbps DL	1000ms for overlay download [Device] -40ms for recognition and file request [Server] -εms for returning file = ~960ms for network
UC2	[1-10]	static to high speed	Worldwide	Assets: [1-10]Mbps UL/DL	500kbps UL 500kbps DL	N/A	N/A	Avatar animation [.5-1]Mbps	100ms user to user [Device] -20ms Audio and facial acquisition and encoding [Device] -20ms audio/avatar decoding = ~60ms for network
UC4	[1-10]	static	Worldwide	360° video [8-16]Mbps UL/DL	500kbps UL 500kbps DL	N/A	N/A	Avatar animation [.5-1]Mbps	300ms room to user to room [Room] -20ms 360 video + audio encoding [User] -20ms decoding [User] -εms composition and display [User] -20 audio/avatar encoding [Room] -20ms audio/avatar decoding [Device] -εms for composition and display =~200ms for the network

In addition to these technical requirements, it is expected from the network to:

- Support temporary boost to cope with punctual application data bursts

5.2.4. Split-rendering for collaborative work

Two use-cases can be deployed using the split-rendering for collaborative work processing scenario:

- UC4: Immersive Telepresence
- UC8: Immersive AR whiteboard

For this processing scenario, the two scenarios require presence to be achieved. In order to reach the expected QoE, and as highlighted in clause 4.5, the interaction between users working on the same common assets should be seamless for the user, which translates in hard latency constraints and support for hard and non-periodic data bursts. In addition, the same expectations as for the split-rendering case apply.

Table 4: Network requirements for split rendering for collaborative work scenario

	Session aspects			Throughput					Latency (worst-case)
	#Users	Device Speed	Distance between users	Video	Audio	Sensors	Haptics	Data	
UC4	[1-10]	static	Worldwide	360° video [8-16]Mbps UL/DL	500kbps UL 500kbps DL	N/A	N/A	Avatar animation [.5-1]Mbps	300ms room to user to room [Room] -20ms volumetric video + audio encoding [User] -1ms Sensor and pose acquisition [User] -20 audio/avatar encoding [Server] -16ms Meeting room scene generation and rendering [Server] -6ms GPU encoding [User] -20ms video decoding (1 frame at 50fps) [User] -εms composition and display [Room] -20ms audio/avatar decoding [Room] -εms composition and display =~150ms for the network
UC8	[1]	Static	N/A	Rendering: 30 Mbps DL	500kbps UL 500kbps DL	Pose 500Kbps UL	500kbps UL 500kbps DL	Assets: [1-30]Mbps UL bursts	100ms motion-to-render-to-photon [Device] -1ms for sensors and pose acquisition [Device] -20ms for acquisition and video encoding (1 frame at 50fps) [Server] -16ms for shared asset update and scene generation [Server] -6ms for GPU encoding [12] [Device] -20ms for decoding (1 frame at 50fps)

									[Device] -ems for composition and display = ~36ms remaining for the network

As for the previous processing scenario, it is also expected from the network to support punctual high throughput/low-latency boost to enable user collaboration.

5.2.5. Split-rendering for remote control with haptics

The last processing scenario is associated with one use-case:

- UC5: Immersive teleoperated driving

This scenario combines aspects from the other scenarios into one, adding stringent requirements on control and haptics transmissions.

Table 5: Network requirements for split rendering for remote control with haptics

	Session aspects			Throughput					Latency (worst-case)
	#Users	Device Speed	Distance between users	Video	Audio	Sensors	Haptics	Data	
UC5	[1]	static driver moving vehicle	Worldwide	360° video [8-16]Mbps From the vehicle to the user overlays [1-10]Mbps	500kbps UL 500kbps DL	[10-20]Mbps from the vehicle to the server	500kps from the vehicle to the user	Control 100kps from the user to the vehicle	120ms end-to-end ([15] [16]) [User] -1ms for control acquisition [vehicle] -20ms audio/video acquisition, haptics and encoding [server] -20ms video decoding [server] -40ms AI/ML + DT merge [server] -6ms video encoding [user] -20ms video decoding [user] -ems for composition and display =~33ms for the network

Similarly to other processing scenarios, additional function requirements are expected from the network(s) to reach a sufficient QoE:

- Flexible throughput management, to adapt bitrate to congestion
- Capability to prioritize sub-flows compared to other ones, based on their relative importance

5.3. Consolidation of requirements and KPIs

In the previous section, various functional, technical requirements and KPIs have been collected for our various processing scenarios. This section provides a consolidation and synthesis of network requirements and capabilities expected to enable the different processing scenarios, thus the various diversity of achievable use-cases. The synthesis of networking requirements is highlighted in the Table below.

Table 6: Network requirements for split rendering for collaborative work scenario

processing scenario	Traffic				Network delay budget	Session
	Component	Data Rate	Traffic type	Packet Error Ratio (PER)		Users per session

Split rendering	DL	Rendered video	[20-30] Mbps	Periodic [30-100] Hz	10^-4 [19-21]	~[17-57]ms (UL+DL) worst case ~[920-960]ms (UL+DL) third person	[1-100]
		Rendered audio	[500] Kbps	Periodic [48] kHz			
		Haptics	[500] Kbps	Periodic [30-100] Hz			
	UL	Pose	[500] Kbps	Periodic [30-100] Hz			
		Application control	[500] Kbps	Non-periodic			
Offloaded overlay generation	DL	Generated overlay	[0.5-10]Mbps	Periodic [30-100] Hz	10^-4 [19-21]	~13ms (UL+DL) with AI/ML ~913ms (UL+DL) for static overlay	[1]
		Generated audio	[500] Kbps	Periodic [48] kHz			
		Text	3 Mb	Punctual burst			
	UL	Pose	[500] Kbps	Periodic [30-100] Hz			
		Camera input	[3-20] Mbps	Periodic [30-100] Hz			
Standalone rendering	DL	A/V assets	[5-30] Mbps	Punctual burst	10^-6 [19-21]	~60ms user to user, worst case ~200ms conference case ~960ms overlay & asset download	[1-100]
		Application assets	[1-5] Mbps	Punctual burst			
		Voice	[500] Kbps	Periodic [48] kHz			
		Avatar animation	[.5-1] Mbps	Periodic [30-100] Hz			
	UL	Voice	[500] Kbps	Periodic [48] kHz			
		Avatar animation	[.5-1] Mbps	Periodic [30-100] Hz			
		Application control	[500] Kbps	Non-periodic			

Split rendering for collaborative work	DL	360° Video	[8-16] Mbps	Periodic [30-100] Hz	10^-3 [19-21]	~150ms for conference ~36ms for 1st person collaborative and interactive scenario	[1-10]
		Rendered video	[20-30] Mbps	Periodic [30-100] Hz			
		Rendered audio	[500] Kbps	Periodic [48] kHz			
		Voice	[500] Kbps	Periodic [48] kHz			
		Haptics	[500] Kbps	[30-100] Hz			
		Data	[.5 30] Mbps	Punctual burst			
	UL	Pose	[500] Kbps	Periodic [30-100] Hz			
		Voice	[500] Kbps	Periodic [48] kHz			
		Haptics	[500] Kbps	Periodic [30-100] Hz			
		Data	[.5 30] Mbps	Punctual burst			
Split-rendering for remote control with haptics (User)	DL	360° Video	[8-16] Mbps	Periodic [30-100] Hz	10^-5 [19-21]	~33ms for ToD	[1]
		Generated overlay	[0.5-10]Mbps	[30-100] Hz			
		Vehicle control	[500] Kbps	Non-periodic			
		Voice	[500] Kbps	Periodic [48] kHz			
		Generated overlay	[0.5-10]Mbps	[30-100] Hz			
	UL	360° Video	[8-16] Mbps	Periodic [30-100] Hz			
		Sensors	[10-20] Mbps	Periodic [30-100] Hz			

		Haptics	[500] Kbps	Periodic [30-100] Hz			

It is observed that the networking technology will have to support more than just stringent connectivity requirements. Indeed, the multi-modal aspect of the traffic and flexibility in its management shall be addressed. This enables to establish the following functional requirements:

Table 7: Consolidated functional network requirements for metaverse processing scenarios

Functional requirements	Split rendering	Offloaded overlay generation	Standalone rendering	Split rendering for collaborative work	Split-rendering for remote control with haptics
Flexible throughput management, to adapt bitrate to signs of congestion	YES	NO	NO	NO	YES
Prioritization of sub-flows compared to other ones, based on their relative importance	YES	NO	NO	NO	YES
Support punctual high-throughput/low-latency bursts	NO	NO	YES	YES	NO

Beside these functional requirements, some network exposure capabilities are required to support deployment of the identified use-cases. The identified network exposure capabilities are the following:

- Quality on Demand to support carriage of media flows
- Network information / network insights to monitor network quality and react upon degradation
- Provisioning/deployment of computing resources at the edge – as close as possible
- Network customization to ensure traffic prioritization for crucial data in case of congestion
- Accurate location and precise positioning information, including indoor, to support real-time services

6. Conclusion

This deliverable provides a list of network functional requirements and KPIs derived from few processing scenarios, driven by end-device capabilities, with KPIs ranges reflecting from the simplest to the most stringent cases. Based on these requirements, their deployment feasibility should be studied, which is the scope of the deliverable #2.

In general, the following requirements are highlighted:

- Network delay budget (RTT) is from 13ms for most demanding applications. In general, 20-25ms would satisfy most of the cases
- When all components are summed up, around 80Mbps-DL/30Mbps-UL are needed in total for most demanding applications. In general, 30Mbps-DL/10Mbps-UL would satisfy most of the cases