|  |
| --- |
| 3GPP TR 26.857 V0.3.1 (2022-08) |
| Technical Report |
| 3rd Generation Partnership Project;Technical Specification Group SA;5G Media Service Enablers(Release 18) |
|   |
|  |  |
|  |
| The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) and may be further elaborated for the purposes of 3GPP.The present document has not been subject to any approval process by the 3GPPOrganizational Partners and shall not be implemented.This Specification is provided for future development work within 3GPPonly. The Organizational Partners accept no liability for any use of this Specification.Specifications and Reports for implementation of the 3GPP TM system should be obtained via the 3GPP Organizational Partners' Publications Offices. |

|  |
| --- |
|  |
| ***3GPP***Postal address3GPP support office address650 Route des Lucioles - Sophia AntipolisValbonne - FRANCETel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16Internethttp://www.3gpp.org |
| ***Copyright Notification***No part may be reproduced except as authorized by written permission.The copyright and the foregoing restriction extend to reproduction in all media.© 2021, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).All rights reserved.UMTS™ is a Trade Mark of ETSI registered for the benefit of its members3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational PartnersLTE™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational PartnersGSM® and the GSM logo are registered and owned by the GSM Association |

Contents

Foreword 4

Introduction 5

1 Scope 6

2 References 6

3 Definitions of terms, symbols and abbreviations 6

3.1 Terms 6

3.2 Symbols 7

3.3 Abbreviations 7

4 Motivation for Media Service Enablers 7

4.1 General 7

4.2 Examples in 3GPP 7

4.2.1 MBMS Client 7

4.2.2 Media Session Handler in 5GMS 10

4.2.3 Media Player in 5GMS 11

4.2.4 SA6 Application Enabler Frameworks 11

4.3 External Specifications 11

4.3.1 W3C HTML-5 APIs for Media 11

4.3.2 Khronos OpenXR 11

5 Considered MSE frameworks 13

5.1 General 13

5.2 MSE framework proposal #1 14

5.2.1 Architecture 14

5.2.2 MSE Specification 14

5.2.3 MSE implementation 15

5.2.4 Example 16

5.3 MSE framework proposal #2 17

5.4 Discussion on different MSE framework proposals 17

6 Properties and Functionalities of MSE 18

7 Tools and Languages for better specifications 19

8 Potentially Relevant 5G Media Service Enablers 19

9 Conclusions and Recommendations 19

Annex <A> (informative): <Normative annex for a Technical Specification> 20

Annex <X> (informative): Change history 22

# Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

In the present document, modal verbs have the following meanings:

**shall** indicates a mandatory requirement to do something

**shall not** indicates an interdiction (prohibition) to do something

The constructions "shall" and "shall not" are confined to the context of normative provisions, and do not appear in Technical Reports.

The constructions "must" and "must not" are not used as substitutes for "shall" and "shall not". Their use is avoided insofar as possible, and they are not used in a normative context except in a direct citation from an external, referenced, non-3GPP document, or so as to maintain continuity of style when extending or modifying the provisions of such a referenced document.

**should** indicates a recommendation to do something

**should not** indicates a recommendation not to do something

**may** indicates permission to do something

**need not** indicates permission not to do something

The construction "may not" is ambiguous and is not used in normative elements. The unambiguous constructions "might not" or "shall not" are used instead, depending upon the meaning intended.

**can** indicates that something is possible

**cannot** indicates that something is impossible

The constructions "can" and "cannot" are not substitutes for "may" and "need not".

**will** indicates that something is certain or expected to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**will not** indicates that something is certain or expected not to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**might** indicates a likelihood that something will happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

**might not** indicates a likelihood that something will not happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

In addition:

**is** (or any other verb in the indicative mood) indicates a statement of fact

**is not** (or any other negative verb in the indicative mood) indicates a statement of fact

The constructions "is" and "is not" do not indicate requirements.

# Introduction

In recent studies and specification work, it was identified that 5G Media functions and 5G System functions need to be made attractive for third-party applications, in particular those that include media delivery. Hence, it is important that these functions are accessible to third-party applications independent of a 3GPP service. For this purpose, it is considered to introduce normative specifications in 3GPP SA4 that are

- more than just a core functionality, e.g. a codec, without any connection to a service or application

- less than a full service that includes all aspects of session establishment, delivery, codecs, rendering and a full user experience

Such new specifications are referred to 5G “Media Service Enablers”.

# 1 Scope

The present document introduces and defines the concept of Media Service Enablers which includes among others:

* Definition of the principal properties of Media Service Enablers.
* Definition of minimum and typical functionalities of Media Service Enablers.
* Definition of a specification template for Media Service Enablers.
* Identification of possibly relevant stage-2 and stage-3 work for Media Service Enablers.
* Collection of a set of initially relevant Media Service Enablers for normative work.

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[ISO-23090-8] ISO/IEC 23090-8:2020: "Information technology — Coded representation of immersive media — Part 8: Network based media processing".

[A] 3GPP TS 26.346: "Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs".

[B] 3GPP TS 26.347: "Multimedia Broadcast/Multicast Service (MBMS); Application Programming Interface and URL".

[C] 3GPP TS 26.479, "UE MBMS APIs for Mission Critical Services".[D] 3GPP TS 26.501, "5G Media Streaming (5GMS); General description and architecture".

[E] 3GPP TS 26.511, "5G Media Streaming (5GMS); Profiles, codecs and formats".

[F] 3GPP TS 26.512, "5G Media Streaming (5GMS); Protocols".

# 3 Definitions of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms given in 3GPP TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in 3GPP TR 21.905 [1].

Definition format (Normal)

**<defined term>:** <definition>.

**example:** text used to clarify abstract rules by applying them literally.

## 3.2 Symbols

For the purposes of the present document, the following symbols apply:

Symbol format (EW)

<symbol> <Explanation>

## 3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in 3GPP TR 21.905 [1].

MSE Media Service Enabler

# 4 Motivation for Media Service Enablers

## 4.1 General

In recent studies and specification work, it was identified that 5G Media functions and 5G System functions need to be made attractive for third-party applications, in particular those that include media delivery. Examples for such approaches are MBMS or 5G Media Streaming. Hence, it is important that these functions are accessible to third-party applications independent of a 3GPP service. For this purpose, it is considered to introduce normative specifications in 3GPP that are:

* More than just a core functionality, e.g. a codec, without any connection to a service or application.
* Less than a full service that includes all aspects of session establishment, delivery, codecs, rendering and a full user experience.

The specification should also not only address a pure textual description but provide additional functionalities such as test and validation tools.

Several examples of specifications at least partially addressing such needs are provided in the remainder of this clause, both 3GPP internal specifications in clause 4.2 and external specification in clause 4.3.

## 4.2 Examples in 3GPP

### 4.2.1 MBMS Client

An example for the definition of an API-centric component in a 3GPP specification is one that serves the MBMS Client. The detailed procedures of the MBMS Client are defined in 3GPP TS 26.346 [A] and TS 26.347 [B] according to Figure 4.2.1-1.



Figure 4.2.1-1: MBMS Client – Application and Network reference Points an APIs

In particular, TS 26.347 defines the following aspects:

1. A set of service APIs for different application user services. The definition provides the ability to independently develop MBMS-Aware Applications and MBMS Client implementations, even for different operating systems and execution environments, but relies on the service APIs to communicate with the MBMS Client and to make use of the MBMS functionalities. These APIs are referred to as MBMS-API-C.

2. A set of interface options between the MBMS Client and the application to support the transfer of user data. The primary focus is on the communication through network interfaces, for example the usage of IP sockets or HTTP-based requests. These APIs are referred to as MBMS-API-U.

Additionally, For Mission Critical (MC) purposes and direct access to MBMS bearer contents, an integration API is specified by the Mission Critical Open Platform [https://www.mcopenplatform.org/]. 3GPP also specifies the MC MBMS API in TS 26.479 [C] based on the same objective.

The APIs defined in TS 26.347 address the following aspects:

- A *client state model* in relation to the application. Examples for state are IDLE, REGISTERED, ACTIVE, etc. State changes may occur through MBMS-API-C or by information received through the network interface.

- A set of *client internal parameters* that are changed based on either configuration or API calls through MBMS-API-C or by information received through the network interface.

- A *reference description* of the operation of the MBMS client in different states, based on through MBMS-API-C or by information received through the network interface

- Different *methods* that allow the application to communicate with the MBMS client. For each method, the following information is provided:

i) A *high-level description* of the method.

ii) An example *call flow* illustrating usage of the method.

iii) A list of input and output *parameters* that are exchanged as part of the method invocation.

iv) A *description of the usage* of the method by the application.

v) the MBMS Client actions in response to the invocation of the method, including pre- and post-conditions.

The equivalent Android APIs for MBMS-API-C are defined in the developer framework of Android:

* Download Session: <https://developer.android.com/reference/android/telephony/MbmsDownloadSession>
* Group Call Session: <https://developer.android.com/reference/android/telephony/MbmsGroupCallSession>
* Streaming Session: <https://developer.android.com/reference/android/telephony/MbmsStreamingSession>
* MBMS API documentation: https://developer.android.com/reference/android/telephony/mbms/package-summary

Finally, TS 26.347 also defines interfaces between the MBMS Client and the application for data exchanges. While the MBMS-API-C provides all methods to find and establish these interfaces, MBMS-API-U provides requirements on the data interfaces, for example for copying files, for requesting files through HTTP, for using specific methods based on an application such as DASH or HLS, or for accessing interfaces that provide RTP packets, UDP datagrams or packet data.

An example usage of the abovementioned Android APIs to support accessing MBMS services through Mission Critical functions is provided as follows using a reception feature activation:

Listing 4.2.1‑1

private MbmsGroupCallSessionCallback groupCallSessionCallback;
private MbmsGroupCallSession mbmsGroupCallSession;
private String mbmsInterfaceName;
*/\*\*
 \* Activate MBMS reception
 \*/*public void enableMBMS() {

 groupCallSessionCallback = new MbmsGroupCallSessionCallback() {

 @Override
 public void onServiceInterfaceAvailable(@NonNull String interfaceName, int index) {
 Log.*e*(*TAG*, "service interface for MBMS Reception " + interfaceName);
 mbmsInterfaceName = interfaceName;
 }

 };

 //Enabling MBMS reception
 mbmsGroupCallSession = MbmsGroupCallSession.*create*(this.getApplicationContext(), 1, this.getMainExecutor(), groupCallSessionCallback);

}

Reception of data from an MBMS bearer is triggered as follows:

Listing 4.2.1‑2

private GroupCall groupCall;

*/\*\*
 \* Starting the reception of a MBMS bearer
 \** ***@param*** *tmgi the Temporary Multicast Group Identifier of the MBMS Bearer
 \*/*public void startReceptionMBMSBearer(long tmgi) {
 GroupCallCallback myCallBack = new GroupCallCallback() {
 @Override
 public void onGroupCallStateChanged(int state, int reason) {
 switch (state) {
 case GroupCall.*STATE\_STARTED*:
 Log.*i*(*TAG*, "MBMS bearer reception is started");
 break;

 case GroupCall.*STATE\_STALLED*:
 Log.*e*(*TAG*, "onGroupCallStateChanged: stalled reason " + reason);
 break;

 case GroupCall.*STATE\_STOPPED*:
 Log.*e*(*TAG*, "onGroupCallStateChanged: stopped reason " + reason);
 break;
 } } };
 //List of Service Area Identifiers and frequencies, may be left empty
 List<Integer> sais = new ArrayList<Integer>();
 List<Integer> frequencies = new ArrayList<Integer>();
 groupCall = mbmsGroupCallSession.startGroupCall(tmgi, sais, frequencies, *executor*, myCallBack);

Finally, the multicast packet data is accessed by the following execution:Listing 4.2.1‑3

*/\*\*
 \* Access to the multicast IP packets
 \** ***@param*** *multicastAddress String representation of the multicast IP address to join
 \** ***@param*** *destinationPort destination port
 \** ***@throws*** *Exception
 \*/*public void receive(String multicastAddress, int destinationPort) throws Exception {
 NetworkInterface ni = NetworkInterface.*getByName*(mInterfaceName);

 //open a multicast socket
 MulticastSocket mSocket = new MulticastSocket(destinationPort);
 SocketAddress socketAddress =
 new InetSocketAddress(multicastAddress, destinationPort);

 //join the multicast group on a given network interface
 mSocket.joinGroup(socketAddress, ni);

 while (true) {
 byte[] buf = new byte[1500];
 DatagramPacket recv = new DatagramPacket(buf, buf.length);
 mSocket.receive(recv);

 //*TODO process the received datagram* }
}

### 4.2.2 Media Session Handler in 5GMS

Editor’s Note:

* We will have a discussion with the 5G-MAG developers on this topic on March 25, 2022
* Can we create background services
	+ Android APIs
	+ What about using HTTP-APIs?
	+ Transparent



### 4.2.3 Media Player in 5GMS

Details tbd

### 4.2.4 SA6 Application Enabler Frameworks

Editor’s Note:

* Rapporteur will speak with SA6 chair on March 25, 2022

## 4.3 External Specifications

### 4.3.1 W3C HTML-5 APIs for Media

EDITOR#S Note.

W3C way

Significant testing environment

2 interoperable implementations

CTA WAVE 5003 Device Playback specification



### 4.3.2 Khronos OpenXR

Android SDKs

* OpenXR SDKs
* API definitions
* Exact ways to write description
* ASCIIDoctor is the document generator tool (see attached)
* Header files are generated
* Validation is generated

RESTful APIs







# 5 Considered MSE frameworks

## 5.1 General

This clause collects some proposed and considered MSE frameworks. A discussion on the different framework proposals is provided in clause 5.4.

## 5.2 MSE framework proposal #1

### 5.2.1 Architecture

Figure 5.2.1-1 shows a possible framework for Media Service Enablers. The MSE framework consists of two parts: the *MSE specification* (on the left of the figure) and the *MSE implementation* (on the right).

**MSE Specification**

(Platform-dependent) **MSE SDK**

Media interfaces

Control interfaces

Configuration API

**MSE SDK abstraction**

Media interfaces

Control interfaces

Configuration API Abstraction

**MSE Service**

Media interfaces

Control interfaces

Configuration API

*Specification*

*Implementation*

MSE Description Document

Media specification

MSE Configuration API

Service API

*Platform-independent*

*Platform-dependent*

Figure 5.2.1-1. Media Service Enablers Framework

### 5.2.2 MSE Specification

An MSE Specification defines:

1. *Media aspects*

a. Functional description of the MSE including the mandatory and optional features.

b. The control interfaces such as provisioning, authentication that is used by the application, and other functions to interact with this MSE.

c. The media interfaces that includes all inputs and outputs format and protocols.

d. Network interface including system and radio network.

e. Event, notifications, reporting, and monitoring.

f. Error handling.

2. *MSE Configuration*

a. An *MSE Description Document (MDD)* that describes:

1. Functions supported by an MSE implementation and their configuration parameters.

2. Optionally the performance/cost metrics for the different features/options.

b. An *MSE Configuration API (MCA)* abstraction for:

1. Retrieving the MSE Description Document.

2. Configuring the MSE instantiation.

3. Retrieving the state and status of the MSE instantiation.

c. A service API for the MSE Configuration API.

Media aspects (1) are usually covered by SA4 specifications. However, the MSE Configuration (2) is absent from current SA4 specifications and is what the MSE Specification adds. The value of this is that, for any SDK or service that is conforming to the MSE specification, a description of the features and their configuration parameters can be retrieved by an external function or service. Additionally, the external function or service can set a specific configuration for running that SDK. Furthermore, the state and status of the running SDK can be retrieved at any time.

The language and syntax of the MSE Description Document and the general framework of the MSE Configuration API can be defined uniformly for all SA4 Media Service Enabler specifications and only specific codepoints are defined in that specification. An external function or application understanding the MSE Description Document syntax, as well as supporting the MSE Configuration API, can retrieve the information from an MSE implementation. If it recognizes the MSE Specification identifier, it can parse and process the MSE Description Document and its configuration parameters.

An example of an MSE Description Document can be found in ISO/IEC 23090-8 [ISO-23090-8]. The function description document is a JSON document that describes the functionalities and features that a function provides as well as its configuration parameters.

### 5.2.3 MSE implementation

An MSE implementation may consist of up to three aspects:

1. The MSE SDK abstraction, an abstract SDK definition intended to be realized as a Software Development Kit, which includes the followings:

i. Media aspects conforming to the MSE specification.

ii. MSE Description Document and MSE Configuration API.

2. The MSE SDK instantiation which is an SDK implementation in a specific environment and conforms to the following:

i. Media aspects conforming to the MSE Specification.

ii. MSE Description Document and a specific implementation of the MSE Configuration API.

3. The MSE service which is the MSE implementation as a service, i.e with APIs that are platform-independent (such as web-based APIs) and conforms to the following:

i. Media aspects conforming to the MSE Specification.

ii. MSE Description Document and a platform-independent implementation of the MSE Configuration API.

As shown in Figure 5.2.1-1, while the MSE SDK abstraction and the MSE Service are platform-independent, the MSE SDK is an instantiation of the MSE SDK abstraction for a specific platform/environment.

An MSE Specification does not require to include all three aspects. For instance, if an MSE is only intended to be realized as software development kit, then its specification would include specifications for the SDK abstraction and one or more SDK instantiation.

### 5.2.4 Example

As shown in figure 5.2.1-1, the MSE Specification can be deployed in two different ways: as an SDK for running on devices or as a microservice running on an Application Server. To demonstrate converting an existing 3GPP specification to an MSE specification, we use the 5GMS Media Session Handler defined in TS 26.501 [D], shown in figure 5.2.4-1.



Figure 5.2.4-1. Media Session Handler as defined in 26.501



Figure 5.2.4-2. Media Session Handler as MSE SDK abstraction, MSE SDK instantiations, and MSE service

The MSE Specification for the Media Session Handler (MSH) shown in Figure 5.2.4-2 describes the following:

1. Media aspects:

a. Functional description of:

i. Service Access Information.

ii. Consumption Reporting.

iii. Metrics Reporting.

iv. Dynamic policies.

v. Network Assistance.

b. M5d, M6d, M7d API definitions:

i. M5d as is already defined.

ii. M6d and M7d as abstract APIs.

iii. M6d and M7d as service APIs.

2. MSE Configuration

a. An MSE Description Document which describes:

i. An identifier that shows this MSE conforms to (1).

ii. Optional features of (1a) and (1b) with their configuration parameters.

iii. Optionally the performance/cost metrics for the different features/options.

b. Abstract API definitions for:

i. Retrieving the MSE Description Document (2a).

ii. Configuring the MSE instantiation.

iii. Retrieving the state and status of the MSE instantiation.

c. A service API for the abstract API (2b).

And MSE SDK implementation of the above specification for Android should support the following:

3. Media aspects conforming to (1), including a specific implementation of the M6d and M7d service APIs.

4. The MSE Description Document (2a) and a specific implementation of the abstract APIs (2b).

The MSE Description Document describes the features implemented by the MSE. The abstract APIs allow an external Android process to retrieve this document and configure the SDK with a set of configurable parameters that are described in the MSE Description Document. They also allow it to interrogate the state and status of the running SDK.

## 5.3 MSE framework proposal #2

## 5.4 Discussion on different MSE framework proposals

# 6 Properties and Functionalities of MSE

Editor’s Note: In implementations and deployments, such packaged functions are typically referred to as Software development kit (SDK) and they are usable by applications through well-defined APIs. A few potential properties of a Media Service Enabler are provided:

* + Set of functions that may be used to develop applications on top of 5G Systems.
	+ Set of robust features and functionalities which reduce the complexity of developing applications
	+ Functions to leverage system and radio optimizations as well as features defined in 5G System (5G Core Network and 5G NR)
	+ Usability of the set of functions by well-defined and well-documented APIs
	+ Provision of network interfaces to connect to the 5G System
	+ A testable set of functions. Testing and conformance may be addressed outside 3GPP by an appropriate MRP or Industry forum.
	+ Guidelines and examples to make use of this set functionalities

A general initial idea on how to define media service enablers are documented below:

* + combine functions defined in 3GPP (for example a codec) and/or may reference technologies defined outside of 3GPP, for example in MPEG or Khronos, and provide relevant subsets and profiles of those
	+ include mandatory, recommended and optional functions. Define signaling and capability negotiation for all functions
	+ specify requirements for client and network functions, as needed
	+ may include relevant functions such as QoE metrics and KPIs

In order to establish the above concept in 3GPP, a clear set of guidelines and requirements for Media Service Enabler specifications is needed.

What may constitute an MSE specification?

* Reference and profile 3GPP specifications and external specifications
* Includes requirements for Codecs & Protocols & Processing & Formats
* Includes capabilities, APIs, reference points, interfaces
* May include QoS Requirements for 5G System

What interfaces are addressed in an MSE specification?

APIs to control the SDK/MSE

* Methods
* Events
* Notifications
* Errors

 Network interfaces

* What terminates/originates in the MSE
* Output to app/display
* Buffers (raw media or encoded media, files)

How should the APIs be designed?

* Should have the following properties
* Should document abstract APIs with semantical descriptions
* APIS can be implemented afterwards
* Directly in Android
* below an existing Android API such that Android can be built on top

What may be properties of an MSE?

* It may have states: idle, active, inactive
* Follow functional requirements
* Possibly even fulfil performance requirements?

What may be documented for an an MSE or an MSE profile?

* Call flows and procedures should be made mandatory
* Methods and APIs
* Pre and post conditions

Use TS 26.347 as a baseline for MSE documentation, but improve on systematics. Reuse some concepts from HTML-5 video element and MSE.

# 7 Tools and Languages for better specifications

Editor’s Note:

What are reasonable abstraction languages?

* YAML/OpenAPIs => RESTFul APIs
* WebIDL
* others

Can we help documentation and specification using more tools?

* MSC
* IDL Editors
* Editing in markdown?

# 8 Potentially Relevant 5G Media Service Enablers

Editor’s Note: collect MSEs that may be defined

# 9 Conclusions and Recommendations

Annex <A> (informative):
<Normative annex for a Technical Specification>

Start each annex on a new page.

Annexes are labelled A, B, C, etc. and designated either "normative" or "informative" depending on their content.

Normative annexes only to appear in Technical Specifications. Use style "Heading 8".

Annex <X> (informative):
Change history

|  |
| --- |
| **Change history** |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
| 2022-02 | SA4#117 | S4-220031 |  |  |  | Initial version | V0.0.1 |
| 2022-02 | SA4#117 | S4-220282 |  |  |  | Version agreed during SA4#117e | V0.1.0 |
| 2022-04 | SA4#118 | S4-220566 |  |  |  | Version agreed during SA4#118e | V0.2.0 |