**3GPP TSG-SA3 Meeting #79-e-b *s3i200723***

**Online, 10-12 Nov 2020**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.0* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **33.128** | **CR** | **0148** | **rev** | **-** | **Current version:** | **16.4.0** |  |
|  | | | | | | | | |
| *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 |  | Radio Access Network |  | Core Network | **x** |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** | Update Serving System for 5G service identifiers (Stage 3) | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | SA3-LI (OTD) | | | | | | | | | |
| ***Source to TSG:*** | SA3 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | LI16 | | | | |  | ***Date:*** | | |  |
|  |  | | | |  | |  | | |  |
| ***Category:*** | **B** |  | | | | | ***Release:*** | | | Rel-16 |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)*  Detailed explanations of the above categories can be found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) Rel-12 (Release 12)* *Rel-13 (Release 13) Rel-14 (Release 14) Rel-15 (Release 15) Rel-16 (Release 16)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | TS 33.128 does not provide an LI capability at the UDM for 5G service identifiers for serving system message. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | Addition of stage three LI events and messages for 5G service identifiers for serving system message. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | No method for the CSP to deliver reporting of 5G enhanced parameters at the UDM. The CSP cannot meet their obligatory requirements for LI. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 7.2.2.3.2, Annex A | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **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:*** | |  | | | | | | | | |

##### 

**\*\*\* Start of First MODIFICATION \*\*\***

##### 7.2.2.3.2 Serving system

The IRI-POI in the UDM shall generate an xIRI containing the UDMServingSystemMessage record when it detects the following events:

- When the UDM receives the amf3GPPAccessRegistration from the AMF in the Nudm\_UEContextManagement\_Registration message (see TS 29.503 [25], clause 5.3.2.2.2).

- When the UDM receives the amfNon3GPPAccessRegistration from the AMF in the Nudm\_UEContextManagement\_Registration message (see TS 29.503 [25], clause 5.3.2.2.3).

- When the UDM receives the amf3GPPAccessRegistration from the MME via the AMF in the Nudm\_UEContextManagement\_Registration message (see TS 23.501 [2], clause 5.17.2.3.2) during inter-system handover.

When a target UE registers to both 3GPP and non-3GPP access, two separate xIRIs each containing the UDMServingSystemMessage record may be generated by the IRI-POI in the UDM.

Table 7.2.2.3-1: Payload for UDMServingSystemMessage record

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| sUPI | SUPI associated with the target UE, see TS 29.571 [17]. | M |
| pEI | PEI associated with the target UE, when known, see TS 29.571 17]. | C |
| gPSI | GPSI associated with the target UE, when known, see TS 29.571 [17]. | C |
| gUAMI | Serving AMF’s GUAMI, when known. See NOTE 1. | C |
| gUMMEI | Serving MME’s GUMMEI See NOTE 2. | C |
| pLMNID | Serving PLMN Id. See TS 29.571 [17]. See NOTE 3. | C |
| servingSystemMethod | Identifies method used to access the serving system, see NOTE 4. | M |
| serviceID | Identifies the target UE’s 5G service identifiers when the AMF Registration is executed, when known, see TS 29.571 [17]. | C |

NOTE 1: GUAMI is the global unique identifier of an AMF [2] and its format is defined in TS 29.571 [17]. As defined in TS 23.501 [2], clause 5.9.4, GUAMI consists of <MCC> <MNC> <AMF Region ID> <AMF Set ID> <AMF Pointer>. The GUAMI is reported if the UDM receives the same from the AMF.

NOTE 2: GUMMEI is the global unique identifier of an MME and its format is defined in TS 23.003 [19]. As defined in TS 23.003 [19], clause 2.8.1, GUMMEI consists of <MCC> <MNC> <MME Identifier>. The GUMMEI is reported if the UDM receives the same from the MME via the AMF.

NOTE 3: PLMN Id provides the VPLMN Id when the target UE is roaming.

NOTE 4: This identifies whether the xIRI containing the UDMServingSystemMessage record is generated due to the reception of an amf3GPPAccessRegistration, or an amfNon3GPPAccessRegistration. See TS 29.503 [25].

TS 29.571 [17] requires that the encoding of 3GPP defined identifiers (e.g. IMSI, NAI) shall be prefixed with its corresponding prefix (e.g. with reference to SUPI it requires 'imsi-','nai-'). However, identifiers and parameters shall be coded over the LI\_X2 and LI\_HI2 according to Annex A of the present document, so without the prefix specified in TS 29.571 [17].

**\*\*\* Start of Thrid MODIFICATION \*\*\***

Annex A (normative): Structure of both the Internal and External Interfaces

TS33128Payloads

{itu-t(0) identified-organization(4) etsi(0) securityDomain(2) lawfulIntercept(2) threeGPP(4) ts33128(19) r16(16) version4(4)}

DEFINITIONS IMPLICIT TAGS EXTENSIBILITY IMPLIED ::=

BEGIN

-- =============

-- Relative OIDs

-- =============

tS33128PayloadsOID RELATIVE-OID ::= {threeGPP(4) ts33128(19) r16(16) version4(4)}

xIRIPayloadOID RELATIVE-OID ::= {tS33128PayloadsOID xIRI(1)}

xCCPayloadOID RELATIVE-OID ::= {tS33128PayloadsOID xCC(2)}

iRIPayloadOID RELATIVE-OID ::= {tS33128PayloadsOID iRI(3)}

cCPayloadOID RELATIVE-OID ::= {tS33128PayloadsOID cC(4)}

lINotificationPayloadOID RELATIVE-OID ::= {tS33128PayloadsOID lINotification(5)}

-- ===============

-- X2 xIRI payload

-- ===============

XIRIPayload ::= SEQUENCE

{

xIRIPayloadOID [1] RELATIVE-OID,

event [2] XIRIEvent

}

XIRIEvent ::= CHOICE

{

-- Access and mobility related events, see clause 6.2.2

registration [1] AMFRegistration,

deregistration [2] AMFDeregistration,

locationUpdate [3] AMFLocationUpdate,

startOfInterceptionWithRegisteredUE [4] AMFStartOfInterceptionWithRegisteredUE,

unsuccessfulAMProcedure [5] AMFUnsuccessfulProcedure,

-- PDU session-related events, see clause 6.2.3

pDUSessionEstablishment [6] SMFPDUSessionEstablishment,

pDUSessionModification [7] SMFPDUSessionModification,

pDUSessionRelease [8] SMFPDUSessionRelease,

startOfInterceptionWithEstablishedPDUSession [9] SMFStartOfInterceptionWithEstablishedPDUSession,

unsuccessfulSMProcedure [10] SMFUnsuccessfulProcedure,

-- Subscriber-management related events, see clause 7.2.2

servingSystemMessage [11] UDMServingSystemMessage,

-- SMS-related events, see clause 6.2.5

sMSMessage [12] SMSMessage,

-- LALS-related events, see clause 7.3.3

lALSReport [13] LALSReport,

-- PDHR/PDSR-related events, see clause 6.2.3.4.1

pDHeaderReport [14] PDHeaderReport,

pDSummaryReport [15] PDSummaryReport,

-- tag 16 is reserved because there is no equivalent mDFCellSiteReport in XIRIEvent

-- MMS-related events, see clause 7.4.2

mMSSend [17] MMSSend,

mMSSendByNonLocalTarget [18] MMSSendByNonLocalTarget,

mMSNotification [19] MMSNotification,

mMSSendToNonLocalTarget [20] MMSSendToNonLocalTarget,

mMSNotificationResponse [21] MMSNotificationResponse,

mMSRetrieval [22] MMSRetrieval,

mMSDeliveryAck [23] MMSDeliveryAck,

mMSForward [24] MMSForward,

mMSDeleteFromRelay [25] MMSDeleteFromRelay,

mMSDeliveryReport [26] MMSDeliveryReport,

mMSDeliveryReportNonLocalTarget [27] MMSDeliveryReportNonLocalTarget,

mMSReadReport [28] MMSReadReport,

mMSReadReportNonLocalTarget [29] MMSReadReportNonLocalTarget,

mMSCancel [30] MMSCancel,

mMSMBoxStore [31] MMSMBoxStore,

mMSMBoxUpload [32] MMSMBoxUpload,

mMSMBoxDelete [33] MMSMBoxDelete,

mMSMBoxViewRequest [34] MMSMBoxViewRequest,

mMSMBoxViewResponse [35] MMSMBoxViewResponse,

-- PTC-related events, see clause 7.5.2

pTCRegistration [36] PTCRegistration,

pTCSessionInitiation [37] PTCSessionInitiation,

pTCSessionAbandon [38] PTCSessionAbandon,

pTCSessionStart [39] PTCSessionStart,

pTCSessionEnd [40] PTCSessionEnd,

pTCStartOfInterception [41] PTCStartOfInterception,

pTCPreEstablishedSession [42] PTCPreEstablishedSession,

pTCInstantPersonalAlert [43] PTCInstantPersonalAlert,

pTCPartyJoin [44] PTCPartyJoin,

pTCPartyDrop [45] PTCPartyDrop,

pTCPartyHold [46] PTCPartyHold,

pTCMediaModification [47] PTCMediaModification,

pTCGroupAdvertisement [48] PTCGroupAdvertisement,

pTCFloorControl [49] PTCFloorControl,

pTCTargetPresence [50] PTCTargetPresence,

pTCParticipantPresence [51] PTCParticipantPresence,

pTCListManagement [52] PTCListManagement,

pTCAccessPolicy [53] PTCAccessPolicy

}

-- ==============

-- X3 xCC payload

-- ==============

-- No additional xCC payload definitions required in the present document.

-- ===============

-- HI2 IRI payload

-- ===============

IRIPayload ::= SEQUENCE

{

iRIPayloadOID [1] RELATIVE-OID,

event [2] IRIEvent,

targetIdentifiers [3] SEQUENCE OF IRITargetIdentifier OPTIONAL

}

IRIEvent ::= CHOICE

{

-- Registration-related events, see clause 6.2.2

registration [1] AMFRegistration,

deregistration [2] AMFDeregistration,

locationUpdate [3] AMFLocationUpdate,

startOfInterceptionWithRegisteredUE [4] AMFStartOfInterceptionWithRegisteredUE,

unsuccessfulRegistrationProcedure [5] AMFUnsuccessfulProcedure,

-- PDU session-related events, see clause 6.2.3

pDUSessionEstablishment [6] SMFPDUSessionEstablishment,

pDUSessionModification [7] SMFPDUSessionModification,

pDUSessionRelease [8] SMFPDUSessionRelease,

startOfInterceptionWithEstablishedPDUSession [9] SMFStartOfInterceptionWithEstablishedPDUSession,

unsuccessfulSessionProcedure [10] SMFUnsuccessfulProcedure,

-- Subscriber-management related events, see clause 7.2.2

servingSystemMessage [11] UDMServingSystemMessage,

-- SMS-related events, see clause 6.2.5

sMSMessage [12] SMSMessage,

-- LALS-related events, see clause 7.3.3

lALSReport [13] LALSReport,

-- PDHR/PDSR-related events, see clause 6.2.3.4.1

pDHeaderReport [14] PDHeaderReport,

pDSummaryReport [15] PDSummaryReport,

-- MDF-related events, see clause 7.3.4

mDFCellSiteReport [16] MDFCellSiteReport,

-- MMS-related events, see clause 7.4.2

mMSSend [17] MMSSend,

mMSSendByNonLocalTarget [18] MMSSendByNonLocalTarget,

mMSNotification [19] MMSNotification,

mMSSendToNonLocalTarget [20] MMSSendToNonLocalTarget,

mMSNotificationResponse [21] MMSNotificationResponse,

mMSRetrieval [22] MMSRetrieval,

mMSDeliveryAck [23] MMSDeliveryAck,

mMSForward [24] MMSForward,

mMSDeleteFromRelay [25] MMSDeleteFromRelay,

mMSDeliveryReport [26] MMSDeliveryReport,

mMSDeliveryReportNonLocalTarget [27] MMSDeliveryReportNonLocalTarget,

mMSReadReport [28] MMSReadReport,

mMSReadReportNonLocalTarget [29] MMSReadReportNonLocalTarget,

mMSCancel [30] MMSCancel,

mMSMBoxStore [31] MMSMBoxStore,

mMSMBoxUpload [32] MMSMBoxUpload,

mMSMBoxDelete [33] MMSMBoxDelete,

mMSMBoxViewRequest [34] MMSMBoxViewRequest,

mMSMBoxViewResponse [35] MMSMBoxViewResponse,

-- PTC-related events, see clause 7.5.2

pTCRegistration [36] PTCRegistration,

pTCSessionInitiation [37] PTCSessionInitiation,

pTCSessionAbandon [38] PTCSessionAbandon,

pTCSessionStart [39] PTCSessionStart,

pTCSessionEnd [40] PTCSessionEnd,

pTCStartOfInterception [41] PTCStartOfInterception,

pTCPreEstablishedSession [42] PTCPreEstablishedSession,

pTCInstantPersonalAlert [43] PTCInstantPersonalAlert,

pTCPartyJoin [44] PTCPartyJoin,

pTCPartyDrop [45] PTCPartyDrop,

pTCPartyHold [46] PTCPartyHold,

pTCMediaModification [47] PTCMediaModification,

pTCGroupAdvertisement [48] PTCGroupAdvertisement,

pTCFloorControl [49] PTCFloorControl,

pTCTargetPresence [50] PTCTargetPresence,

pTCParticipantPresence [51] PTCParticipantPresence,

pTCListManagement [52] PTCListManagement,

pTCAccessPolicy [53] PTCAccessPolicy

}

}

IRITargetIdentifier ::= SEQUENCE

{

identifier [1] TargetIdentifier,

provenance [2] TargetIdentifierProvenance OPTIONAL

}

-- ==============

-- HI3 CC payload

-- ==============

CCPayload ::= SEQUENCE

{

cCPayloadOID [1] RELATIVE-OID,

pDU [2] CCPDU

}

CCPDU ::= CHOICE

{

uPFCCPDU [1] UPFCCPDU,

extendedUPFCCPDU [2] ExtendedUPFCCPDU,

mMSCCPDU [3] MMSCCPDU

}

-- ===========================

-- HI4 LI notification payload

-- ===========================

LINotificationPayload ::= SEQUENCE

{

lINotificationPayloadOID [1] RELATIVE-OID,

notification [2] LINotificationMessage

}

LINotificationMessage ::= CHOICE

{

lINotification [1] LINotification

}

-- ==================

-- 5G AMF definitions

-- ==================

-- See clause 6.2.2.2.2 for details of this structure

AMFRegistration ::= SEQUENCE

{

registrationType [1] AMFRegistrationType,

registrationResult [2] AMFRegistrationResult,

slice [3] Slice OPTIONAL,

sUPI [4] SUPI,

sUCI [5] SUCI OPTIONAL,

pEI [6] PEI OPTIONAL,

gPSI [7] GPSI OPTIONAL,

gUTI [8] FiveGGUTI,

location [9] Location OPTIONAL,

non3GPPAccessEndpoint [10] UEEndpointAddress OPTIONAL

}

-- See clause 6.2.2.2.3 for details of this structure

AMFDeregistration ::= SEQUENCE

{

deregistrationDirection [1] AMFDirection,

accessType [2] AccessType,

sUPI [3] SUPI OPTIONAL,

sUCI [4] SUCI OPTIONAL,

pEI [5] PEI OPTIONAL,

gPSI [6] GPSI OPTIONAL,

gUTI [7] FiveGGUTI OPTIONAL,

cause [8] FiveGMMCause OPTIONAL,

location [9] Location OPTIONAL

}

-- See clause 6.2.2.2.4 for details of this structure

AMFLocationUpdate ::= SEQUENCE

{

sUPI [1] SUPI,

sUCI [2] SUCI OPTIONAL,

pEI [3] PEI OPTIONAL,

gPSI [4] GPSI OPTIONAL,

gUTI [5] FiveGGUTI OPTIONAL,

location [6] Location

}

-- See clause 6.2.2.2.5 for details of this structure

AMFStartOfInterceptionWithRegisteredUE ::= SEQUENCE

{

registrationResult [1] AMFRegistrationResult,

registrationType [2] AMFRegistrationType OPTIONAL,

slice [3] Slice OPTIONAL,

sUPI [4] SUPI,

sUCI [5] SUCI OPTIONAL,

pEI [6] PEI OPTIONAL,

gPSI [7] GPSI OPTIONAL,

gUTI [8] FiveGGUTI,

location [9] Location OPTIONAL,

non3GPPAccessEndpoint [10] UEEndpointAddress OPTIONAL,

timeOfRegistration [11] Timestamp OPTIONAL

}

-- See clause 6.2.2.2.6 for details of this structure

AMFUnsuccessfulProcedure ::= SEQUENCE

{

failedProcedureType [1] AMFFailedProcedureType,

failureCause [2] AMFFailureCause,

requestedSlice [3] NSSAI OPTIONAL,

sUPI [4] SUPI OPTIONAL,

sUCI [5] SUCI OPTIONAL,

pEI [6] PEI OPTIONAL,

gPSI [7] GPSI OPTIONAL,

gUTI [8] FiveGGUTI OPTIONAL,

location [9] Location OPTIONAL

}

-- =================

-- 5G AMF parameters

-- =================

AMFID ::= SEQUENCE

{

aMFRegionID [1] AMFRegionID,

aMFSetID [2] AMFSetID,

aMFPointer [3] AMFPointer

}

AMFDirection ::= ENUMERATED

{

networkInitiated(1),

uEInitiated(2)

}

AMFFailedProcedureType ::= ENUMERATED

{

registration(1),

sMS(2),

pDUSessionEstablishment(3)

}

AMFFailureCause ::= CHOICE

{

fiveGMMCause [1] FiveGMMCause,

fiveGSMCause [2] FiveGSMCause

}

AMFPointer ::= INTEGER (0..63)

AMFRegistrationResult ::= ENUMERATED

{

threeGPPAccess(1),

nonThreeGPPAccess(2),

threeGPPAndNonThreeGPPAccess(3)

}

AMFRegionID ::= INTEGER (0..255)

AMFRegistrationType ::= ENUMERATED

{

initial(1),

mobility(2),

periodic(3),

emergency(4)

}

AMFSetID ::= INTEGER (0..1023)

-- ==================

-- 5G SMF definitions

-- ==================

-- See clause 6.2.3.2.2 for details of this structure

SMFPDUSessionEstablishment ::= SEQUENCE

{

sUPI [1] SUPI OPTIONAL,

sUPIUnauthenticated [2] SUPIUnauthenticatedIndication OPTIONAL,

pEI [3] PEI OPTIONAL,

gPSI [4] GPSI OPTIONAL,

pDUSessionID [5] PDUSessionID,

gTPTunnelID [6] FTEID,

pDUSessionType [7] PDUSessionType,

sNSSAI [8] SNSSAI OPTIONAL,

uEEndpoint [9] SEQUENCE OF UEEndpointAddress OPTIONAL,

non3GPPAccessEndpoint [10] UEEndpointAddress OPTIONAL,

location [11] Location OPTIONAL,

dNN [12] DNN,

aMFID [13] AMFID OPTIONAL,

hSMFURI [14] HSMFURI OPTIONAL,

requestType [15] FiveGSMRequestType,

accessType [16] AccessType OPTIONAL,

rATType [17] RATType OPTIONAL,

sMPDUDNRequest [18] SMPDUDNRequest OPTIONAL

}

-- See clause 6.2.3.2.3 for details of this structure

SMFPDUSessionModification ::= SEQUENCE

{

sUPI [1] SUPI OPTIONAL,

sUPIUnauthenticated [2] SUPIUnauthenticatedIndication OPTIONAL,

pEI [3] PEI OPTIONAL,

gPSI [4] GPSI OPTIONAL,

sNSSAI [5] SNSSAI OPTIONAL,

non3GPPAccessEndpoint [6] UEEndpointAddress OPTIONAL,

location [7] Location OPTIONAL,

requestType [8] FiveGSMRequestType,

accessType [9] AccessType OPTIONAL,

rATType [10] RATType OPTIONAL

}

-- See clause 6.2.3.2.4 for details of this structure

SMFPDUSessionRelease ::= SEQUENCE

{

sUPI [1] SUPI,

pEI [2] PEI OPTIONAL,

gPSI [3] GPSI OPTIONAL,

pDUSessionID [4] PDUSessionID,

timeOfFirstPacket [5] Timestamp OPTIONAL,

timeOfLastPacket [6] Timestamp OPTIONAL,

uplinkVolume [7] INTEGER OPTIONAL,

downlinkVolume [8] INTEGER OPTIONAL,

location [9] Location OPTIONAL

}

-- See clause 6.2.3.2.5 for details of this structure

SMFStartOfInterceptionWithEstablishedPDUSession ::= SEQUENCE

{

sUPI [1] SUPI OPTIONAL,

sUPIUnauthenticated [2] SUPIUnauthenticatedIndication OPTIONAL,

pEI [3] PEI OPTIONAL,

gPSI [4] GPSI OPTIONAL,

pDUSessionID [5] PDUSessionID,

gTPTunnelID [6] FTEID,

pDUSessionType [7] PDUSessionType,

sNSSAI [8] SNSSAI OPTIONAL,

uEEndpoint [9] SEQUENCE OF UEEndpointAddress,

non3GPPAccessEndpoint [10] UEEndpointAddress OPTIONAL,

location [11] Location OPTIONAL,

dNN [12] DNN,

aMFID [13] AMFID OPTIONAL,

hSMFURI [14] HSMFURI OPTIONAL,

requestType [15] FiveGSMRequestType,

accessType [16] AccessType OPTIONAL,

rATType [17] RATType OPTIONAL,

sMPDUDNRequest [18] SMPDUDNRequest OPTIONAL

}

-- See clause 6.2.3.2.6 for details of this structure

SMFUnsuccessfulProcedure ::= SEQUENCE

{

failedProcedureType [1] SMFFailedProcedureType,

failureCause [2] FiveGSMCause,

initiator [3] Initiator,

requestedSlice [4] NSSAI OPTIONAL,

sUPI [5] SUPI OPTIONAL,

sUPIUnauthenticated [6] SUPIUnauthenticatedIndication OPTIONAL,

pEI [7] PEI OPTIONAL,

gPSI [8] GPSI OPTIONAL,

pDUSessionID [9] PDUSessionID OPTIONAL,

uEEndpoint [10] SEQUENCE OF UEEndpointAddress OPTIONAL,

non3GPPAccessEndpoint [11] UEEndpointAddress OPTIONAL,

dNN [12] DNN OPTIONAL,

aMFID [13] AMFID OPTIONAL,

hSMFURI [14] HSMFURI OPTIONAL,

requestType [15] FiveGSMRequestType OPTIONAL,

accessType [16] AccessType OPTIONAL,

rATType [17] RATType OPTIONAL,

sMPDUDNRequest [18] SMPDUDNRequest OPTIONAL,

location [19] Location OPTIONAL

}

-- =================

-- 5G SMF parameters

-- =================

SMFFailedProcedureType ::= ENUMERATED

{

pDUSessionEstablishment(1),

pDUSessionModification(2),

pDUSessionRelease(3)

}

-- ==================

-- 5G UPF definitions

-- ==================

UPFCCPDU ::= OCTET STRING

-- See clause 6.2.3.8 for the details of this structure

ExtendedUPFCCPDU ::= SEQUENCE

{

payload [1] UPFCCPDUPayload,

qFI [2] QFI OPTIONAL

}

-- =================

-- 5G UPF parameters

-- =================

UPFCCPDUPayload ::= CHOICE

{

uPFIPCC [1] OCTET STRING,

uPFEthernetCC [2] OCTET STRING,

uPFUnstructuredCC [3] OCTET STRING

}

QFI ::= INTEGER (0..63)

-- ==================

-- 5G UDM definitions

-- ==================

UDMServingSystemMessage ::= SEQUENCE

{

sUPI [1] SUPI,

pEI [2] PEI OPTIONAL,

gPSI [3] GPSI OPTIONAL,

gUAMI [4] GUAMI OPTIONAL,

gUMMEI [5] GUMMEI OPTIONAL,

pLMNID [6] PLMNID OPTIONAL,

servingSystemMethod [7] UDMServingSystemMethod,

serviceID [8] ServiceID OPTIONAL

}

-- =================

-- 5G UDM parameters

-- =================

UDMServingSystemMethod ::= ENUMERATED

{

amf3GPPAccessRegistration(0),

amfNon3GPPAccessRegistration(1),

unknown(2)

}

ServiceID ::= SEQUENCE

{

sNSSAI [1] SNSSAI,

cAGID [2] OCTET STRING (SIZE (4))

}

-- ===================

-- 5G SMSF definitions

-- ===================

-- See clause 6.2.5.3 for details of this structure

SMSMessage ::= SEQUENCE

{

originatingSMSParty [1] SMSParty,

terminatingSMSParty [2] SMSParty,

direction [3] Direction,

transferStatus [4] SMSTransferStatus,

otherMessage [5] SMSOtherMessageIndication OPTIONAL,

location [6] Location OPTIONAL,

peerNFAddress [7] SMSNFAddress OPTIONAL,

peerNFType [8] SMSNFType OPTIONAL,

sMSTPDUData [9] SMSTPDUData OPTIONAL

}

-- ==================

-- 5G SMSF parameters

-- ==================

SMSParty ::= SEQUENCE

{

sUPI [1] SUPI OPTIONAL,

pEI [2] PEI OPTIONAL,

gPSI [3] GPSI OPTIONAL

}

SMSTransferStatus ::= ENUMERATED

{

transferSucceeded(1),

transferFailed(2),

undefined(3)

}

SMSOtherMessageIndication ::= BOOLEAN

SMSNFAddress ::= CHOICE

{

iPAddress [1] IPAddress,

e164Number [2] E164Number

}

SMSNFType ::= ENUMERATED

{

sMSGMSC(1),

iWMSC(2),

sMSRouter(3)

}

SMSTPDUData ::= CHOICE

{

sMSTPDU [1] SMSTPDU

}

SMSTPDU ::= OCTET STRING (SIZE(1..270))

-- ===============

-- MMS definitions

-- ===============

MMSSend ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

dateTime [3] Timestamp,

originatingMMSParty [4] MMSParty,

terminatingMMSParty [5] SEQUENCE OF MMSParty OPTIONAL,

cCRecipients [6] SEQUENCE OF MMSParty OPTIONAL,

bCCRecipients [7] SEQUENCE OF MMSParty OPTIONAL,

direction [8] MMSDirection,

subject [9] MMSSubject OPTIONAL,

messageClass [10] MMSMessageClass OPTIONAL,

expiry [11] MMSExpiry,

desiredDeliveryTime [12] Timestamp OPTIONAL,

priority [13] MMSPriority OPTIONAL,

senderVisibility [14] BOOLEAN OPTIONAL,

deliveryReport [15] BOOLEAN OPTIONAL,

readReport [16] BOOLEAN OPTIONAL,

store [17] BOOLEAN OPTIONAL,

state [18] MMState OPTIONAL,

flags [19] MMFlags OPTIONAL,

replyCharging [20] MMSReplyCharging OPTIONAL,

applicID [21] UTF8String OPTIONAL,

replyApplicID [22] UTF8String OPTIONAL,

auxApplicInfo [23] UTF8String OPTIONAL,

contentClass [24] MMSContentClass OPTIONAL,

dRMContent [25] BOOLEAN OPTIONAL,

adaptationAllowed [26] MMSAdaptation OPTIONAL,

contentType [27] MMSContentType,

responseStatus [28] MMSResponseStatus,

responseStatusText [29] UTF8String OPTIONAL,

messageID [30] UTF8String

}

MMSSendByNonLocalTarget ::= SEQUENCE

{

version [1] MMSVersion,

transactionID [2] UTF8String,

messageID [3] UTF8String,

terminatingMMSParty [4] SEQUENCE OF MMSParty,

originatingMMSParty [5] MMSParty,

direction [6] MMSDirection,

contentType [7] MMSContentType,

messageClass [8] MMSMessageClass OPTIONAL,

dateTime [9] Timestamp,

expiry [10] MMSExpiry OPTIONAL,

deliveryReport [11] BOOLEAN OPTIONAL,

priority [12] MMSPriority OPTIONAL,

senderVisibility [13] BOOLEAN OPTIONAL,

readReport [14] BOOLEAN OPTIONAL,

subject [15] MMSSubject OPTIONAL,

forwardCount [16] INTEGER OPTIONAL,

previouslySentBy [17] MMSPreviouslySentBy OPTIONAL,

prevSentByDateTime [18] Timestamp OPTIONAL,

applicID [19] UTF8String OPTIONAL,

replyApplicID [20] UTF8String OPTIONAL,

auxApplicInfo [21] UTF8String OPTIONAL,

contentClass [22] MMSContentClass OPTIONAL,

dRMContent [23] BOOLEAN OPTIONAL,

adaptationAllowed [24] MMSAdaptation OPTIONAL

}

MMSNotification ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

originatingMMSParty [3] MMSParty OPTIONAL,

direction [4] MMSDirection,

subject [5] MMSSubject OPTIONAL,

deliveryReportRequested [6] BOOLEAN OPTIONAL,

stored [7] BOOLEAN OPTIONAL,

messageClass [8] MMSMessageClass,

priority [9] MMSPriority OPTIONAL,

messageSize [10] INTEGER,

expiry [11] MMSExpiry,

replyCharging [12] MMSReplyCharging OPTIONAL

}

MMSSendToNonLocalTarget ::= SEQUENCE

{

version [1] MMSVersion,

transactionID [2] UTF8String,

messageID [3] UTF8String,

terminatingMMSParty [4] SEQUENCE OF MMSParty,

originatingMMSParty [5] MMSParty,

direction [6] MMSDirection,

contentType [7] MMSContentType,

messageClass [8] MMSMessageClass OPTIONAL,

dateTime [9] Timestamp,

expiry [10] MMSExpiry OPTIONAL,

deliveryReport [11] BOOLEAN OPTIONAL,

priority [12] MMSPriority OPTIONAL,

senderVisibility [13] BOOLEAN OPTIONAL,

readReport [14] BOOLEAN OPTIONAL,

subject [15] MMSSubject OPTIONAL,

forwardCount [16] INTEGER OPTIONAL,

previouslySentBy [17] MMSPreviouslySentBy OPTIONAL,

prevSentByDateTime [18] Timestamp OPTIONAL,

applicID [19] UTF8String OPTIONAL,

replyApplicID [20] UTF8String OPTIONAL,

auxApplicInfo [21] UTF8String OPTIONAL,

contentClass [22] MMSContentClass OPTIONAL,

dRMContent [23] BOOLEAN OPTIONAL,

adaptationAllowed [24] MMSAdaptation OPTIONAL

}

MMSNotificationResponse ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

direction [3] MMSDirection,

status [4] MMStatus,

reportAllowed [5] BOOLEAN OPTIONAL

}

MMSRetrieval ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

messageID [3] UTF8String,

dateTime [4] Timestamp,

originatingMMSParty [5] MMSParty OPTIONAL,

previouslySentBy [6] MMSPreviouslySentBy OPTIONAL,

prevSentByDateTime [7] Timestamp OPTIONAL,

terminatingMMSParty [8] SEQUENCE OF MMSParty OPTIONAL,

cCRecipients [9] SEQUENCE OF MMSParty OPTIONAL,

direction [10] MMSDirection,

subject [11] MMSSubject OPTIONAL,

state [12] MMState OPTIONAL,

flags [13] MMFlags OPTIONAL,

messageClass [14] MMSMessageClass OPTIONAL,

priority [15] MMSPriority,

deliveryReport [16] BOOLEAN OPTIONAL,

readReport [17] BOOLEAN OPTIONAL,

replyCharging [18] MMSReplyCharging OPTIONAL,

retrieveStatus [19] MMSRetrieveStatus OPTIONAL,

retrieveStatusText [20] UTF8String OPTIONAL,

applicID [21] UTF8String OPTIONAL,

replyApplicID [22] UTF8String OPTIONAL,

auxApplicInfo [23] UTF8String OPTIONAL,

contentClass [24] MMSContentClass OPTIONAL,

dRMContent [25] BOOLEAN OPTIONAL,

replaceID [26] UTF8String OPTIONAL,

contentType [27] UTF8String OPTIONAL

}

MMSDeliveryAck ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

reportAllowed [3] BOOLEAN OPTIONAL,

status [4] MMStatus,

direction [5] MMSDirection

}

MMSForward ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

dateTime [3] Timestamp OPTIONAL,

originatingMMSParty [4] MMSParty,

terminatingMMSParty [5] SEQUENCE OF MMSParty OPTIONAL,

cCRecipients [6] SEQUENCE OF MMSParty OPTIONAL,

bCCRecipients [7] SEQUENCE OF MMSParty OPTIONAL,

direction [8] MMSDirection,

expiry [9] MMSExpiry OPTIONAL,

desiredDeliveryTime [10] Timestamp OPTIONAL,

deliveryReportAllowed [11] BOOLEAN OPTIONAL,

deliveryReport [12] BOOLEAN OPTIONAL,

store [13] BOOLEAN OPTIONAL,

state [14] MMState OPTIONAL,

flags [15] MMFlags OPTIONAL,

contentLocationReq [16] UTF8String,

replyCharging [17] MMSReplyCharging OPTIONAL,

responseStatus [18] MMSResponseStatus,

responseStatusText [19] UTF8String OPTIONAL,

messageID [20] UTF8String OPTIONAL,

contentLocationConf [21] UTF8String OPTIONAL,

storeStatus [22] MMSStoreStatus OPTIONAL,

storeStatusText [23] UTF8String OPTIONAL

}

MMSDeleteFromRelay ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

direction [3] MMSDirection,

contentLocationReq [4] SEQUENCE OF UTF8String,

contentLocationConf [5] SEQUENCE OF UTF8String,

deleteResponseStatus [6] MMSDeleteResponseStatus,

deleteResponseText [7] SEQUENCE OF UTF8String

}

MMSMBoxStore ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

direction [3] MMSDirection,

contentLocationReq [4] UTF8String,

state [5] MMState OPTIONAL,

flags [6] MMFlags OPTIONAL,

contentLocationConf [7] UTF8String OPTIONAL,

storeStatus [8] MMSStoreStatus,

storeStatusText [9] UTF8String OPTIONAL

}

MMSMBoxUpload ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

direction [3] MMSDirection,

state [4] MMState OPTIONAL,

flags [5] MMFlags OPTIONAL,

contentType [6] UTF8String,

contentLocation [7] UTF8String OPTIONAL,

storeStatus [8] MMSStoreStatus,

storeStatusText [9] UTF8String OPTIONAL,

mMessages [10] SEQUENCE OF MMBoxDescription

}

MMSMBoxDelete ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

direction [3] MMSDirection,

contentLocationReq [4] SEQUENCE OF UTF8String,

contentLocationConf [5] SEQUENCE OF UTF8String OPTIONAL,

responseStatus [6] MMSDeleteResponseStatus,

responseStatusText [7] UTF8String OPTIONAL

}

MMSDeliveryReport ::= SEQUENCE

{

version [1] MMSVersion,

messageID [2] UTF8String,

terminatingMMSParty [3] SEQUENCE OF MMSParty,

mMSDateTime [4] Timestamp,

responseStatus [5] MMSResponseStatus,

responseStatusText [6] UTF8String OPTIONAL,

applicID [7] UTF8String OPTIONAL,

replyApplicID [8] UTF8String OPTIONAL,

auxApplicInfo [9] UTF8String OPTIONAL

}

MMSDeliveryReportNonLocalTarget ::= SEQUENCE

{

version [1] MMSVersion,

transactionID [2] UTF8String,

messageID [3] UTF8String,

terminatingMMSParty [4] SEQUENCE OF MMSParty,

originatingMMSParty [5] MMSParty,

direction [6] MMSDirection,

mMSDateTime [7] Timestamp,

forwardToOriginator [8] BOOLEAN OPTIONAL,

status [9] MMStatus,

statusExtension [10] MMStatusExtension,

statusText [11] MMStatusText,

applicID [12] UTF8String OPTIONAL,

replyApplicID [13] UTF8String OPTIONAL,

auxApplicInfo [14] UTF8String OPTIONAL

}

MMSReadReport ::= SEQUENCE

{

version [1] MMSVersion,

messageID [2] UTF8String,

terminatingMMSParty [3] SEQUENCE OF MMSParty,

originatingMMSParty [4] SEQUENCE OF MMSParty,

direction [5] MMSDirection,

mMSDateTime [6] Timestamp,

readStatus [7] MMSReadStatus,

applicID [8] UTF8String OPTIONAL,

replyApplicID [9] UTF8String OPTIONAL,

auxApplicInfo [10] UTF8String OPTIONAL

}

MMSReadReportNonLocalTarget ::= SEQUENCE

{

version [1] MMSVersion,

transactionID [2] UTF8String,

terminatingMMSParty [3] SEQUENCE OF MMSParty,

originatingMMSParty [4] SEQUENCE OF MMSParty,

direction [5] MMSDirection,

messageID [6] UTF8String,

mMSDateTime [7] Timestamp,

readStatus [8] MMSReadStatus,

readStatusText [9] MMSReadStatusText OPTIONAL,

applicID [10] UTF8String OPTIONAL,

replyApplicID [11] UTF8String OPTIONAL,

auxApplicInfo [12] UTF8String OPTIONAL

}

MMSCancel ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

cancelID [3] UTF8String,

direction [4] MMSDirection

}

MMSMBoxViewRequest ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

contentLocation [3] UTF8String OPTIONAL,

state [4] SEQUENCE OF MMState OPTIONAL,

flags [5] SEQUENCE OF MMFlags OPTIONAL,

start [6] INTEGER OPTIONAL,

limit [7] INTEGER OPTIONAL,

attributes [8] SEQUENCE OF UTF8String OPTIONAL,

totals [9] INTEGER OPTIONAL,

quotas [10] MMSQuota OPTIONAL

}

MMSMBoxViewResponse ::= SEQUENCE

{

transactionID [1] UTF8String,

version [2] MMSVersion,

contentLocation [3] UTF8String OPTIONAL,

state [4] SEQUENCE OF MMState OPTIONAL,

flags [5] SEQUENCE OF MMFlags OPTIONAL,

start [6] INTEGER OPTIONAL,

limit [7] INTEGER OPTIONAL,

attributes [8] SEQUENCE OF UTF8String OPTIONAL,

mMSTotals [9] BOOLEAN OPTIONAL,

mMSQuotas [10] BOOLEAN OPTIONAL,

mMessages [11] SEQUENCE OF MMBoxDescription

}

MMBoxDescription ::= SEQUENCE

{

contentLocation [1] UTF8String OPTIONAL,

messageID [2] UTF8String OPTIONAL,

state [3] MMState OPTIONAL,

flags [4] SEQUENCE OF MMFlags OPTIONAL,

dateTime [5] Timestamp OPTIONAL,

originatingMMSParty [6] MMSParty OPTIONAL,

terminatingMMSParty [7] SEQUENCE OF MMSParty OPTIONAL,

cCRecipients [8] SEQUENCE OF MMSParty OPTIONAL,

bCCRecipients [9] SEQUENCE OF MMSParty OPTIONAL,

messageClass [10] MMSMessageClass OPTIONAL,

subject [11] MMSSubject OPTIONAL,

priority [12] MMSPriority OPTIONAL,

deliveryTime [13] Timestamp OPTIONAL,

readReport [14] BOOLEAN OPTIONAL,

messageSize [15] INTEGER OPTIONAL,

replyCharging [16] MMSReplyCharging OPTIONAL,

previouslySentBy [17] MMSPreviouslySentBy OPTIONAL,

previouslySentByDateTime [18] Timestamp OPTIONAL,

contentType [19] UTF8String OPTIONAL

}

-- =========

-- MMS CCPDU

-- =========

MMSCCPDU ::= SEQUENCE

{

version [1] MMSVersion,

transactionID [2] UTF8String,

mMSContent [3] OCTET STRING

}

-- ==============

-- MMS parameters

-- ==============

MMSAdaptation ::= SEQUENCE

{

allowed [1] BOOLEAN,

overriden [2] BOOLEAN

}

MMSCancelStatus ::= ENUMERATED

{

cancelRequestSuccessfullyReceived(1),

cancelRequestCorrupted(2)

}

MMSContentClass ::= ENUMERATED

{

text(1),

imageBasic(2),

imageRich(3),

videoBasic(4),

videoRich(5),

megaPixel(6),

contentBasic(7),

contentRich(8)

}

MMSContentType ::= UTF8String

MMSDeleteResponseStatus ::= ENUMERATED

{

ok(1),

errorUnspecified(2),

errorServiceDenied(3),

errorMessageFormatCorrupt(4),

errorSendingAddressUnresolved(5),

errorMessageNotFound(6),

errorNetworkProblem(7),

errorContentNotAccepted(8),

errorUnsupportedMessage(9),

errorTransientFailure(10),

errorTransientSendingAddressUnresolved(11),

errorTransientMessageNotFound(12),

errorTransientNetworkProblem(13),

errorTransientPartialSuccess(14),

errorPermanentFailure(15),

errorPermanentServiceDenied(16),

errorPermanentMessageFormatCorrupt(17),

errorPermanentSendingAddressUnresolved(18),

errorPermanentMessageNotFound(19),

errorPermanentContentNotAccepted(20),

errorPermanentReplyChargingLimitationsNotMet(21),

errorPermanentReplyChargingRequestNotAccepted(22),

errorPermanentReplyChargingForwardingDenied(23),

errorPermanentReplyChargingNotSupported(24),

errorPermanentAddressHidingNotSupported(25),

errorPermanentLackOfPrepaid(26)

}

MMSDirection ::= ENUMERATED

{

fromTarget(0),

toTarget(1)

}

MMSElementDescriptor ::= SEQUENCE

{

reference [1] UTF8String,

parameter [2] UTF8String OPTIONAL,

value [3] UTF8String OPTIONAL

}

MMSExpiry ::= SEQUENCE

{

expiryPeriod [1] INTEGER,

periodFormat [2] MMSPeriodFormat

}

MMFlags ::= SEQUENCE

{

length [1] INTEGER,

flag [2] MMStateFlag,

flagString [3] UTF8String

}

MMSMessageClass ::= ENUMERATED

{

personal(1),

advertisement(2),

informational(3),

auto(4)

}

MMSParty ::= SEQUENCE

{

mMSPartyIDs [1] SEQUENCE OF MMSPartyID,

nonLocalID [2] NonLocalID

}

MMSPartyID ::= CHOICE

{

e164Number [1] E164Number,

emailAddress [2] EmailAddress,

iMSI [3] IMSI,

iMPU [4] IMPU,

iMPI [5] IMPI,

sUPI [6] SUPI,

gPSI [7] GPSI

}

MMSPeriodFormat ::= ENUMERATED

{

absolute(1),

relative(2)

}

MMSPreviouslySent ::= SEQUENCE

{

previouslySentByParty [1] MMSParty,

sequenceNumber [2] INTEGER,

previousSendDateTime [3] Timestamp

}

MMSPreviouslySentBy ::= SEQUENCE OF MMSPreviouslySent

MMSPriority ::= ENUMERATED

{

low(1),

normal(2),

high(3)

}

MMSQuota ::= SEQUENCE

{

quota [1] INTEGER,

quotaUnit [2] MMSQuotaUnit

}

MMSQuotaUnit ::= ENUMERATED

{

numMessages(1),

bytes(2)

}

MMSReadStatus ::= ENUMERATED

{

read(1),

deletedWithoutBeingRead(2)

}

MMSReadStatusText ::= UTF8String

MMSReplyCharging ::= ENUMERATED

{

requested(0),

requestedTextOnly(1),

accepted(2),

acceptedTextOnly(3)

}

MMSResponseStatus ::= ENUMERATED

{

ok(1),

errorUnspecified(2),

errorServiceDenied(3),

errorMessageFormatCorrupt(4),

errorSendingAddressUnresolved(5),

errorMessageNotFound(6),

errorNetworkProblem(7),

errorContentNotAccepted(8),

errorUnsupportedMessage(9),

errorTransientFailure(10),

errorTransientSendingAddressUnresolved(11),

errorTransientMessageNotFound(12),

errorTransientNetworkProblem(13),

errorTransientPartialSuccess(14),

errorPermanentFailure(15),

errorPermanentServiceDenied(16),

errorPermanentMessageFormatCorrupt(17),

errorPermanentSendingAddressUnresolved(18),

errorPermanentMessageNotFound(19),

errorPermanentContentNotAccepted(20),

errorPermanentReplyChargingLimitationsNotMet(21),

errorPermanentReplyChargingRequestNotAccepted(22),

errorPermanentReplyChargingForwardingDenied(23),

errorPermanentReplyChargingNotSupported(24),

errorPermanentAddressHidingNotSupported(25),

errorPermanentLackOfPrepaid(26)

}

MMSRetrieveStatus ::= ENUMERATED

{

success(1),

errorTransientFailure(2),

errorTransientMessageNotFound(3),

errorTransientNetworkProblem(4),

errorPermanentFailure(5),

errorPermanentServiceDenied(6),

errorPermanentMessageNotFound(7),

errorPermanentContentUnsupported(8)

}

MMSStoreStatus ::= ENUMERATED

{

success(1),

errorTransientFailure(2),

errorTransientNetworkProblem(3),

errorPermanentFailure(4),

errorPermanentServiceDenied(5),

errorPermanentMessageFormatCorrupt(6),

errorPermanentMessageNotFound(7),

errorMMBoxFull(8)

}

MMState ::= ENUMERATED

{

draft(1),

sent(2),

new(3),

retrieved(4),

forwarded(5)

}

MMStateFlag ::= ENUMERATED

{

add(1),

remove(2),

filter(3)

}

MMStatus ::= ENUMERATED

{

expired(1),

retrieved(2),

rejected(3),

deferred(4),

unrecognized(5),

indeterminate(6),

forwarded(7),

unreachable(8)

}

MMStatusExtension ::= ENUMERATED

{

rejectionByMMSRecipient(0),

rejectionByOtherRS(1)

}

MMStatusText ::= UTF8String

MMSSubject ::= UTF8String

MMSVersion ::= SEQUENCE

{

majorVersion [1] INTEGER,

minorVersion [2] INTEGER

}

-- ==================

-- 5G PTC definitions

-- ==================

PTCRegistration ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCServerURI [2] UTF8String,

pTCRegistrationRequest [3] PTCRegistrationRequest,

pTCRegistrationOutcome [4] PTCRegistrationOutcome

}

PTCSessionInitiation ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCServerURI [3] UTF8String,

pTCSessionInfo [4] PTCSessionInfo,

pTCOriginatingID [5] PTCTargetInformation,

pTCParticipants [6] SEQUENCE OF PTCTargetInformation OPTIONAL,

pTCParticipantPresenceStatus [7] MultipleParticipantPresenceStatus OPTIONAL,

location [8] Location OPTIONAL,

pTCBearerCapability [9] UTF8String OPTIONAL,

pTCHost [10] PTCTargetInformation OPTIONAL

}

PTCSessionAbandon ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCSessionInfo [3] PTCSessionInfo,

location [4] Location OPTIONAL,

pTCAbandonCause [5] INTEGER

}

PTCSessionStart ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCServerURI [3] UTF8String,

pTCSessionInfo [4] PTCSessionInfo,

pTCOriginatingID [5] PTCTargetInformation,

pTCParticipants [6] SEQUENCE OF PTCTargetInformation OPTIONAL,

pTCParticipantPresenceStatus [7] MultipleParticipantPresenceStatus OPTIONAL,

location [8] Location OPTIONAL,

pTCHost [9] PTCTargetInformation OPTIONAL,

pTCBearerCapability [10] UTF8String OPTIONAL

}

PTCSessionEnd ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCServerURI [3] UTF8String,

pTCSessionInfo [4] PTCSessionInfo,

pTCParticipants [5] SEQUENCE OF PTCTargetInformation OPTIONAL,

location [6] Location OPTIONAL,

pTCSessionEndCause [7] PTCSessionEndCause

}

PTCStartOfInterception ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

preEstSessionID [3] PTCSessionInfo OPTIONAL,

pTCOriginatingID [4] PTCTargetInformation,

pTCSessionInfo [5] PTCSessionInfo OPTIONAL,

pTCHost [6] PTCTargetInformation OPTIONAL,

pTCParticipants [7] SEQUENCE OF PTCTargetInformation OPTIONAL,

pTCMediaStreamAvail [8] BOOLEAN OPTIONAL,

pTCBearerCapability [9] UTF8String OPTIONAL

}

PTCPreEstablishedSession ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCServerURI [2] UTF8String,

rTPSetting [3] RTPSetting,

pTCMediaCapability [4] UTF8String,

pTCPreEstSessionID [5] PTCSessionInfo,

pTCPreEstStatus [6] PTCPreEstStatus,

pTCMediaStreamAvail [7] BOOLEAN OPTIONAL,

location [8] Location OPTIONAL,

pTCFailureCode [9] PTCFailureCode OPTIONAL

}

PTCInstantPersonalAlert ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCIPAPartyID [2] PTCTargetInformation,

pTCIPADirection [3] Direction

}

PTCPartyJoin ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCSessionInfo [3] PTCSessionInfo,

pTCParticipants [4] SEQUENCE OF PTCTargetInformation OPTIONAL,

pTCParticipantPresenceStatus [5] MultipleParticipantPresenceStatus OPTIONAL,

pTCMediaStreamAvail [6] BOOLEAN OPTIONAL,

pTCBearerCapability [7] UTF8String OPTIONAL

}

PTCPartyDrop ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCSessionInfo [3] PTCSessionInfo,

pTCPartyDrop [4] PTCTargetInformation,

pTCParticipantPresenceStatus [5] PTCParticipantPresenceStatus OPTIONAL

}

PTCPartyHold ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCSessionInfo [3] PTCSessionInfo,

pTCParticipants [4] SEQUENCE OF PTCTargetInformation OPTIONAL,

pTCHoldID [5] SEQUENCE OF PTCTargetInformation,

pTCHoldRetrieveInd [6] BOOLEAN

}

PTCMediaModification ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCSessionInfo [3] PTCSessionInfo,

pTCMediaStreamAvail [4] BOOLEAN OPTIONAL,

pTCBearerCapability [5] UTF8String

}

PTCGroupAdvertisement ::=SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCIDList [3] SEQUENCE OF PTCTargetInformation OPTIONAL,

pTCGroupAuthRule [4] PTCGroupAuthRule OPTIONAL,

pTCGroupAdSender [5] PTCTargetInformation,

pTCGroupNickname [6] UTF8String OPTIONAL

}

PTCFloorControl ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCSessioninfo [3] PTCSessionInfo,

pTCFloorActivity [4] SEQUENCE OF PTCFloorActivity,

pTCFloorSpeakerID [5] PTCTargetInformation OPTIONAL,

pTCMaxTBTime [6] INTEGER OPTIONAL,

pTCQueuedFloorControl [7] BOOLEAN OPTIONAL,

pTCQueuedPosition [8] INTEGER OPTIONAL,

pTCTalkBurstPriority [9] PTCTBPriorityLevel OPTIONAL,

pTCTalkBurstReason [10] PTCTBReasonCode OPTIONAL

}

PTCTargetPresence ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCTargetPresenceStatus [2] PTCParticipantPresenceStatus

}

PTCParticipantPresence ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCParticipantPresenceStatus [2] PTCParticipantPresenceStatus

}

PTCListManagement ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCListManagementType [3] PTCListManagementType OPTIONAL,

pTCListManagementAction [4] PTCListManagementAction OPTIONAL,

pTCListManagementFailure [5] PTCListManagementFailure OPTIONAL,

pTCContactID [6] PTCTargetInformation OPTIONAL,

pTCIDList [7] SEQUENCE OF PTCIDList OPTIONAL,

pTCHost [8] PTCTargetInformation OPTIONAL

}

PTCAccessPolicy ::= SEQUENCE

{

pTCTargetInformation [1] PTCTargetInformation,

pTCDirection [2] Direction,

pTCAccessPolicyType [3] PTCAccessPolicyType OPTIONAL,

pTCUserAccessPolicy [4] PTCUserAccessPolicy OPTIONAL,

pTCGroupAuthRule [5] PTCGroupAuthRule OPTIONAL,

pTCContactID [6] PTCTargetInformation OPTIONAL,

pTCAccessPolicyFailure [7] PTCAccessPolicyFailure OPTIONAL

}

-- =================

-- 5G PTC parameters

-- =================

PTCRegistrationRequest ::= ENUMERATED

{

register(1),

reRegister(2),

deRegister(3)

}

PTCRegistrationOutcome ::= ENUMERATED

{

success(1),

failure(2)

}

PTCSessionEndCause ::= ENUMERATED

{

initiaterLeavesSession(1),

definedParticipantLeaves(2),

numberOfParticipants(3),

sessionTimerExpired(4),

pTCSpeechInactive(5),

allMediaTypesInactive(6)

}

PTCTargetInformation ::= SEQUENCE

{

identifiers [1] SEQUENCE SIZE(1..MAX) OF PTCIdentifiers

}

PTCIdentifiers ::= CHOICE

{

mCPTTID [1] UTF8String,

instanceIdentifierURN [2] UTF8String,

pTCChatGroupID [3] PTCChatGroupID,

iMPU [4] IMPU,

iMPI [5] IMPI

}

PTCSessionInfo ::= SEQUENCE

{

pTCSessionURI [1] UTF8String,

pTCSessionType [2] PTCSessionType

}

PTCSessionType ::= ENUMERATED

{

ondemand(1),

preEstablished(2),

adhoc(3),

prearranged(4),

groupSession(5)

}

MultipleParticipantPresenceStatus ::= SEQUENCE OF PTCParticipantPresenceStatus

PTCParticipantPresenceStatus ::= SEQUENCE

{

presenceID [1] PTCTargetInformation,

presenceType [2] PTCPresenceType,

presenceStatus [3] BOOLEAN

}

PTCPresenceType ::= ENUMERATED

{

pTCClient(1),

pTCGroup(2)

}

PTCPreEstStatus ::= ENUMERATED

{

established(1),

modified(2),

released(3)

}

RTPSetting ::= SEQUENCE

{

iPAddress [1] IPAddress,

portNumber [2] PortNumber

}

PTCIDList ::= SEQUENCE

{

pTCPartyID [1] PTCTargetInformation,

pTCChatGroupID [2] PTCChatGroupID

}

PTCChatGroupID ::= SEQUENCE

{

groupIdentity [1] UTF8String

}

PTCFloorActivity ::= ENUMERATED

{

tBCPRequest(1),

tBCPGranted(2),

tBCPDeny(3),

tBCPIdle(4),

tBCPTaken(5),

tBCPRevoke(6),

tBCPQueued(7),

tBCPRelease(8)

}

PTCTBPriorityLevel ::= ENUMERATED

{

preEmptive(1),

highPriority(2),

normalPriority(3),

listenOnly(4)

}

PTCTBReasonCode ::= ENUMERATED

{

noQueuingAllowed(1),

oneParticipantSession(2),

listenOnly(3),

exceededMaxDuration(4),

tBPrevented(5)

}

PTCListManagementType ::= ENUMERATED

{

contactListManagementAttempt(1),

groupListManagementAttempt(2),

contactListManagementResult(3),

groupListManagementResult(4),

requestUnsuccessful(5)

}

PTCListManagementAction ::= ENUMERATED

{

create(1),

modify(2),

retrieve(3),

delete(4),

notify(5)

}

PTCAccessPolicyType ::= ENUMERATED

{

pTCUserAccessPolicyAttempt(1),

groupAuthorizationRulesAttempt(2),

pTCUserAccessPolicyQuery(3),

groupAuthorizationRulesQuery(4),

pTCUserAccessPolicyResult(5),

groupAuthorizationRulesResult(6),

requestUnsuccessful(7)

}

PTCUserAccessPolicy ::= ENUMERATED

{

allowIncomingPTCSessionRequest(1),

blockIncomingPTCSessionRequest(2),

allowAutoAnswerMode(3),

allowOverrideManualAnswerMode(4)

}

PTCGroupAuthRule ::= ENUMERATED

{

allowInitiatingPTCSession(1),

blockInitiatingPTCSession(2),

allowJoiningPTCSession(3),

blockJoiningPTCSession(4),

allowAddParticipants(5),

blockAddParticipants(6),

allowSubscriptionPTCSessionState(7),

blockSubscriptionPTCSessionState(8),

allowAnonymity(9),

forbidAnonymity(10)

}

PTCFailureCode ::= ENUMERATED

{

sessionCannotBeEstablished(1),

sessionCannotBeModified(2)

}

PTCListManagementFailure ::= ENUMERATED

{

requestUnsuccessful(1),

requestUnknown(2)

}

PTCAccessPolicyFailure ::= ENUMERATED

{

requestUnsuccessful(1),

requestUnknown(2)

}

-- ===================

-- 5G LALS definitions

-- ===================

LALSReport ::= SEQUENCE

{

sUPI [1] SUPI OPTIONAL,

pEI [2] PEI OPTIONAL,

gPSI [3] GPSI OPTIONAL,

location [4] Location OPTIONAL

}

-- =====================

-- PDHR/PDSR definitions

-- =====================

PDHeaderReport ::= SEQUENCE

{

pDUSessionID [1] PDUSessionID,

sourceIPAddress [2] IPAddress,

sourcePort [3] PortNumber OPTIONAL,

destinationIPAddress [4] IPAddress,

destinationPort [5] PortNumber OPTIONAL,

nextLayerProtocol [6] NextLayerProtocol,

iPv6flowLabel [7] IPv6FlowLabel OPTIONAL,

direction [8] Direction,

packetSize [9] INTEGER

}

PDSummaryReport ::= SEQUENCE

{

pDUSessionID [1] PDUSessionID,

sourceIPAddress [2] IPAddress,

sourcePort [3] PortNumber OPTIONAL,

destinationIPAddress [4] IPAddress,

destinationPort [5] PortNumber OPTIONAL,

nextLayerProtocol [6] NextLayerProtocol,

iPv6flowLabel [7] IPv6FlowLabel OPTIONAL,

direction [8] Direction,

pDSRSummaryTrigger [9] PDSRSummaryTrigger,

firstPacketTimestamp [10] Timestamp,

lastPacketTimestamp [11] Timestamp,

packetCount [12] INTEGER,

byteCount [13] INTEGER

}

-- ====================

-- PDHR/PDSR parameters

-- ====================

PDSRSummaryTrigger ::= ENUMERATED

{

timerExpiry(1),

packetCount(2),

byteCount(3)

}

-- ===========================

-- LI Notification definitions

-- ===========================

LINotification ::= SEQUENCE

{

notificationType [1] LINotificationType,

appliedTargetID [2] TargetIdentifier OPTIONAL,

appliedDeliveryInformation [3] SEQUENCE OF LIAppliedDeliveryInformation OPTIONAL,

appliedStartTime [4] Timestamp OPTIONAL,

appliedEndTime [5] Timestamp OPTIONAL

}

-- ==========================

-- LI Notification parameters

-- ==========================

LINotificationType ::= ENUMERATED

{

activation(1),

deactivation(2),

modification(3)

}

LIAppliedDeliveryInformation ::= SEQUENCE

{

hI2DeliveryIPAddress [1] IPAddress OPTIONAL,

hI2DeliveryPortNumber [2] PortNumber OPTIONAL,

hI3DeliveryIPAddress [3] IPAddress OPTIONAL,

hI3DeliveryPortNumber [4] PortNumber OPTIONAL

}

-- ===============

-- MDF definitions

-- ===============

MDFCellSiteReport ::= SEQUENCE OF CellInformation

-- =================

-- Common Parameters

-- =================

AccessType ::= ENUMERATED

{

threeGPPAccess(1),

nonThreeGPPAccess(2),

threeGPPandNonThreeGPPAccess(3)

}

Direction ::= ENUMERATED

{

fromTarget(1),

toTarget(2)

}

DNN ::= UTF8String

E164Number ::= NumericString (SIZE(1..15))

EmailAddress ::= UTF8String

FiveGGUTI ::= SEQUENCE

{

mCC [1] MCC,

mNC [2] MNC,

aMFRegionID [3] AMFRegionID,

aMFSetID [4] AMFSetID,

aMFPointer [5] AMFPointer,

fiveGTMSI [6] FiveGTMSI

}

FiveGMMCause ::= INTEGER (0..255)

FiveGSMRequestType ::= ENUMERATED

{

initialRequest(1),

existingPDUSession(2),

initialEmergencyRequest(3),

existingEmergencyPDUSession(4),

modificationRequest(5),

reserved(6),

mAPDURequest(7)

}

FiveGSMCause ::= INTEGER (0..255)

FiveGTMSI ::= INTEGER (0..4294967295)

FTEID ::= SEQUENCE

{

tEID [1] INTEGER (0.. 4294967295),

iPv4Address [2] IPv4Address OPTIONAL,

iPv6Address [3] IPv6Address OPTIONAL

}

GPSI ::= CHOICE

{

mSISDN [1] MSISDN,

nAI [2] NAI

}

GUAMI ::= SEQUENCE

{

aMFID [1] AMFID,

pLMNID [2] PLMNID

}

GUMMEI ::= SEQUENCE

{

mMEID [1] MMEID,

mCC [2] MCC,

mNC [3] MNC

}

HomeNetworkPublicKeyID ::= OCTET STRING

HSMFURI ::= UTF8String

IMEI ::= NumericString (SIZE(14))

IMEISV ::= NumericString (SIZE(16))

IMPI ::= NAI

IMPU ::= CHOICE

{

sIPURI [1] SIPURI,

tELURI [2] TELURI

}

IMSI ::= NumericString (SIZE(6..15))

Initiator ::= ENUMERATED

{

uE(1),

network(2),

unknown(3)

}

IPAddress ::= CHOICE

{

iPv4Address [1] IPv4Address,

iPv6Address [2] IPv6Address

}

IPv4Address ::= OCTET STRING (SIZE(4))

IPv6Address ::= OCTET STRING (SIZE(16))

IPv6FlowLabel ::= INTEGER(0..1048575)

MACAddress ::= OCTET STRING (SIZE(6))

MCC ::= NumericString (SIZE(3))

MNC ::= NumericString (SIZE(2..3))

MMEID ::= SEQUENCE

{

mMEGI [1] MMEGI,

mMEC [2] MMEC

}

MMEC ::= NumericString

MMEGI ::= NumericString

MSISDN ::= NumericString (SIZE(1..15))

NAI ::= UTF8String

NextLayerProtocol ::= INTEGER(0..255)

NonLocalID ::= ENUMERATED

{

local(1),

nonLocal(2)

}

NSSAI ::= SEQUENCE OF SNSSAI

PLMNID ::= SEQUENCE

{

mCC [1] MCC,

mNC [2] MNC

}

PDUSessionID ::= INTEGER (0..255)

PDUSessionType ::= ENUMERATED

{

iPv4(1),

iPv6(2),

iPv4v6(3),

unstructured(4),

ethernet(5)

}

PEI ::= CHOICE

{

iMEI [1] IMEI,

iMEISV [2] IMEISV

}

PortNumber ::= INTEGER(0..65535)

ProtectionSchemeID ::= INTEGER (0..15)

RATType ::= ENUMERATED

{

nR(1),

eUTRA(2),

wLAN(3),

virtual(4),

nBIOT(5),

wireline(6),

wirelineCable(7),

wirelineBBF(8),

lTEM(9),

nRU(10),

eUTRAU(11),

trustedN3GA(12),

trustedWLAN(13),

uTRA(14),

gERA(15)

}

RejectedNSSAI ::= SEQUENCE OF RejectedSNSSAI

RejectedSNSSAI ::= SEQUENCE

{

causeValue [1] RejectedSliceCauseValue,

sNSSAI [2] SNSSAI

}

RejectedSliceCauseValue ::= INTEGER (0..255)

RoutingIndicator ::= INTEGER (0..9999)

SchemeOutput ::= OCTET STRING

SIPURI ::= UTF8String

Slice ::= SEQUENCE

{

allowedNSSAI [1] NSSAI OPTIONAL,

configuredNSSAI [2] NSSAI OPTIONAL,

rejectedNSSAI [3] RejectedNSSAI OPTIONAL

}

SMPDUDNRequest ::= OCTET STRING

SNSSAI ::= SEQUENCE

{

sliceServiceType [1] INTEGER (0..255),

sliceDifferentiator [2] OCTET STRING (SIZE(3)) OPTIONAL

}

SUCI ::= SEQUENCE

{

mCC [1] MCC,

mNC [2] MNC,

routingIndicator [3] RoutingIndicator,

protectionSchemeID [4] ProtectionSchemeID,

homeNetworkPublicKeyID [5] HomeNetworkPublicKeyID,

schemeOutput [6] SchemeOutput

}

SUPI ::= CHOICE

{

iMSI [1] IMSI,

nAI [2] NAI

}

SUPIUnauthenticatedIndication ::= BOOLEAN

TargetIdentifier ::= CHOICE

{

sUPI [1] SUPI,

iMSI [2] IMSI,

pEI [3] PEI,

iMEI [4] IMEI,

gPSI [5] GPSI,

mSISDN [6] MSISDN,

nAI [7] NAI,

iPv4Address [8] IPv4Address,

iPv6Address [9] IPv6Address,

ethernetAddress [10] MACAddress

}

TargetIdentifierProvenance ::= ENUMERATED

{

lEAProvided(1),

observed(2),

matchedOn(3),

other(4)

}

TELURI ::= UTF8String

Timestamp ::= GeneralizedTime

UEEndpointAddress ::= CHOICE

{

iPv4Address [1] IPv4Address,

iPv6Address [2] IPv6Address,

ethernetAddress [3] MACAddress

}

-- ===================

-- Location parameters

-- ===================

Location ::= SEQUENCE

{

locationInfo [1] LocationInfo OPTIONAL,

positioningInfo [2] PositioningInfo OPTIONAL,

locationPresenceReport [3] LocationPresenceReport OPTIONAL

}

CellSiteInformation ::= SEQUENCE

{

geographicalCoordinates [1] GeographicalCoordinates,

azimuth [2] INTEGER (0..359) OPTIONAL,

operatorSpecificInformation [3] UTF8String OPTIONAL

}

-- TS 29.518 [22], clause 6.4.6.2.6

LocationInfo ::= SEQUENCE

{

userLocation [1] UserLocation OPTIONAL,

currentLoc [2] BOOLEAN OPTIONAL,

geoInfo [3] GeographicArea OPTIONAL,

rATType [4] RATType OPTIONAL,

timeZone [5] TimeZone OPTIONAL,

additionalCellIDs [6] SEQUENCE OF CellInformation OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.7

UserLocation ::= SEQUENCE

{

eUTRALocation [1] EUTRALocation OPTIONAL,

nRLocation [2] NRLocation OPTIONAL,

n3GALocation [3] N3GALocation OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.8

EUTRALocation ::= SEQUENCE

{

tAI [1] TAI,

eCGI [2] ECGI,

ageOfLocatonInfo [3] INTEGER OPTIONAL,

uELocationTimestamp [4] Timestamp OPTIONAL,

geographicalInformation [5] UTF8String OPTIONAL,

geodeticInformation [6] UTF8String OPTIONAL,

globalNGENbID [7] GlobalRANNodeID OPTIONAL,

cellSiteInformation [8] CellSiteInformation OPTIONAL,

globalENbID [9] GlobalRANNodeID OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.9

NRLocation ::= SEQUENCE

{

tAI [1] TAI,

nCGI [2] NCGI,

ageOfLocatonInfo [3] INTEGER OPTIONAL,

uELocationTimestamp [4] Timestamp OPTIONAL,

geographicalInformation [5] UTF8String OPTIONAL,

geodeticInformation [6] UTF8String OPTIONAL,

globalGNbID [7] GlobalRANNodeID OPTIONAL,

cellSiteInformation [8] CellSiteInformation OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.10

N3GALocation ::= SEQUENCE

{

tAI [1] TAI OPTIONAL,

n3IWFID [2] N3IWFIDNGAP OPTIONAL,

uEIPAddr [3] IPAddr OPTIONAL,

portNumber [4] INTEGER OPTIONAL

}

-- TS 38.413 [23], clause 9.3.2.4

IPAddr ::= SEQUENCE

{

iPv4Addr [1] IPv4Address OPTIONAL,

iPv6Addr [2] IPv6Address OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.28

GlobalRANNodeID ::= SEQUENCE

{

pLMNID [1] PLMNID,

aNNodeID [2] ANNodeID,

nID [3] NID OPTIONAL

}

ANNodeID ::= CHOICE

{

n3IWFID [1] N3IWFIDSBI,

gNbID [2] GNbID,

nGENbID [3] NGENbID,

eNbID [4] ENbID

}

-- TS 38.413 [23], clause 9.3.1.6

GNbID ::= BIT STRING(SIZE(22..32))

-- TS 29.571 [17], clause 5.4.4.4

TAI ::= SEQUENCE

{

pLMNID [1] PLMNID,

tAC [2] TAC,

nID [3] NID OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.5

ECGI ::= SEQUENCE

{

pLMNID [1] PLMNID,

eUTRACellID [2] EUTRACellID,

nID [3] NID OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.6

NCGI ::= SEQUENCE

{

pLMNID [1] PLMNID,

nRCellID [2] NRCellID,

nID [3] NID OPTIONAL

}

RANCGI ::= CHOICE

{

eCGI [1] ECGI,

nCGI [2] NCGI

}

CellInformation ::= SEQUENCE

{

rANCGI [1] RANCGI,

cellSiteinformation [2] CellSiteInformation OPTIONAL,

timeOfLocation [3] Timestamp OPTIONAL

}

-- TS 38.413 [23], clause 9.3.1.57

N3IWFIDNGAP ::= BIT STRING (SIZE(16))

-- TS 29.571 [17], clause 5.4.4.28

N3IWFIDSBI ::= UTF8String

-- TS 29.571 [17], table 5.4.2-1

TAC ::= OCTET STRING (SIZE(2..3))

-- TS 38.413 [23], clause 9.3.1.9

EUTRACellID ::= BIT STRING (SIZE(28))

-- TS 38.413 [23], clause 9.3.1.7

NRCellID ::= BIT STRING (SIZE(36))

-- TS 38.413 [23], clause 9.3.1.8

NGENbID ::= CHOICE

{

macroNGENbID [1] BIT STRING (SIZE(20)),

shortMacroNGENbID [2] BIT STRING (SIZE(18)),

longMacroNGENbID [3] BIT STRING (SIZE(21))

}

-- TS 23.003 [19], clause 12.7.1 encoded as per TS 29.571 [17], clause 5.4.2

NID ::= UTF8String (SIZE(11))

-- TS 36.413 [38], clause 9.2.1.37

ENbID ::= CHOICE

{

macroENbID [1] BIT STRING (SIZE(20)),

homeENbID [2] BIT STRING (SIZE(28)),

shortMacroENbID [3] BIT STRING (SIZE(18)),

longMacroENbID [4] BIT STRING (SIZE(21))

}

-- TS 29.518 [22], clause 6.4.6.2.3

PositioningInfo ::= SEQUENCE

{

positionInfo [1] LocationData OPTIONAL,

rawMLPResponse [2] RawMLPResponse OPTIONAL

}

RawMLPResponse ::= CHOICE

{

-- The following parameter contains a copy of unparsed XML code of the

-- MLP response message, i.e. the entire XML document containing

-- a <slia> (described in OMA-TS-MLP-V3\_5-20181211-C [20], clause 5.2.3.2.2) or

-- a <slirep> (described in OMA-TS-MLP-V3\_5-20181211-C [20], clause 5.2.3.2.3) MLP message.

mLPPositionData [1] UTF8String,

-- OMA MLP result id, defined in OMA-TS-MLP-V3\_5-20181211-C [20], Clause 5.4

mLPErrorCode [2] INTEGER (1..699)

}

-- TS 29.572 [24], clause 6.1.6.2.3

LocationData ::= SEQUENCE

{

locationEstimate [1] GeographicArea,

accuracyFulfilmentIndicator [2] AccuracyFulfilmentIndicator OPTIONAL,

ageOfLocationEstimate [3] AgeOfLocationEstimate OPTIONAL,

velocityEstimate [4] VelocityEstimate OPTIONAL,

civicAddress [5] CivicAddress OPTIONAL,

positioningDataList [6] SET OF PositioningMethodAndUsage OPTIONAL,

gNSSPositioningDataList [7] SET OF GNSSPositioningMethodAndUsage OPTIONAL,

eCGI [8] ECGI OPTIONAL,

nCGI [9] NCGI OPTIONAL,

altitude [10] Altitude OPTIONAL,

barometricPressure [11] BarometricPressure OPTIONAL

}

-- TS 29.518 [22], clause 6.2.6.2.5

LocationPresenceReport ::= SEQUENCE

{

type [1] AMFEventType,

timestamp [2] Timestamp,

areaList [3] SET OF AMFEventArea OPTIONAL,

timeZone [4] TimeZone OPTIONAL,

accessTypes [5] SET OF AccessType OPTIONAL,

rMInfoList [6] SET OF RMInfo OPTIONAL,

cMInfoList [7] SET OF CMInfo OPTIONAL,

reachability [8] UEReachability OPTIONAL,

location [9] UserLocation OPTIONAL,

additionalCellIDs [10] SEQUENCE OF CellInformation OPTIONAL

}

-- TS 29.518 [22], clause 6.2.6.3.3

AMFEventType ::= ENUMERATED

{

locationReport(1),

presenceInAOIReport(2)

}

-- TS 29.518 [22], clause 6.2.6.2.16

AMFEventArea ::= SEQUENCE

{

presenceInfo [1] PresenceInfo OPTIONAL,

lADNInfo [2] LADNInfo OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.27

PresenceInfo ::= SEQUENCE

{

presenceState [1] PresenceState OPTIONAL,

trackingAreaList [2] SET OF TAI OPTIONAL,

eCGIList [3] SET OF ECGI OPTIONAL,

nCGIList [4] SET OF NCGI OPTIONAL,

globalRANNodeIDList [5] SET OF GlobalRANNodeID OPTIONAL,

globalENbIDList [6] SET OF GlobalRANNodeID OPTIONAL

}

-- TS 29.518 [22], clause 6.2.6.2.17

LADNInfo ::= SEQUENCE

{

lADN [1] UTF8String,

presence [2] PresenceState OPTIONAL

}

-- TS 29.571 [17], clause 5.4.3.20

PresenceState ::= ENUMERATED

{

inArea(1),

outOfArea(2),

unknown(3),

inactive(4)

}

-- TS 29.518 [22], clause 6.2.6.2.8

RMInfo ::= SEQUENCE

{

rMState [1] RMState,

accessType [2] AccessType

}

-- TS 29.518 [22], clause 6.2.6.2.9

CMInfo ::= SEQUENCE

{

cMState [1] CMState,

accessType [2] AccessType

}

-- TS 29.518 [22], clause 6.2.6.3.7

UEReachability ::= ENUMERATED

{

unreachable(1),

reachable(2),

regulatoryOnly(3)

}

-- TS 29.518 [22], clause 6.2.6.3.9

RMState ::= ENUMERATED

{

registered(1),

deregistered(2)

}

-- TS 29.518 [22], clause 6.2.6.3.10

CMState ::= ENUMERATED

{

idle(1),

connected(2)

}

-- TS 29.572 [24], clause 6.1.6.2.5

GeographicArea ::= CHOICE

{

point [1] Point,

pointUncertaintyCircle [2] PointUncertaintyCircle,

pointUncertaintyEllipse [3] PointUncertaintyEllipse,

polygon [4] Polygon,

pointAltitude [5] PointAltitude,

pointAltitudeUncertainty [6] PointAltitudeUncertainty,

ellipsoidArc [7] EllipsoidArc

}

-- TS 29.572 [24], clause 6.1.6.3.12

AccuracyFulfilmentIndicator ::= ENUMERATED

{

requestedAccuracyFulfilled(1),

requestedAccuracyNotFulfilled(2)

}

-- TS 29.572 [24], clause 6.1.6.2.17

VelocityEstimate ::= CHOICE

{

horVelocity [1] HorizontalVelocity,

horWithVertVelocity [2] HorizontalWithVerticalVelocity,

horVelocityWithUncertainty [3] HorizontalVelocityWithUncertainty,

horWithVertVelocityAndUncertainty [4] HorizontalWithVerticalVelocityAndUncertainty

}

-- TS 29.572 [24], clause 6.1.6.2.14

CivicAddress ::= SEQUENCE

{

country [1] UTF8String,

a1 [2] UTF8String OPTIONAL,

a2 [3] UTF8String OPTIONAL,

a3 [4] UTF8String OPTIONAL,

a4 [5] UTF8String OPTIONAL,

a5 [6] UTF8String OPTIONAL,

a6 [7] UTF8String OPTIONAL,

prd [8] UTF8String OPTIONAL,

pod [9] UTF8String OPTIONAL,

sts [10] UTF8String OPTIONAL,

hno [11] UTF8String OPTIONAL,

hns [12] UTF8String OPTIONAL,

lmk [13] UTF8String OPTIONAL,

loc [14] UTF8String OPTIONAL,

nam [15] UTF8String OPTIONAL,

pc [16] UTF8String OPTIONAL,

bld [17] UTF8String OPTIONAL,

unit [18] UTF8String OPTIONAL,

flr [19] UTF8String OPTIONAL,

room [20] UTF8String OPTIONAL,

plc [21] UTF8String OPTIONAL,

pcn [22] UTF8String OPTIONAL,

pobox [23] UTF8String OPTIONAL,

addcode [24] UTF8String OPTIONAL,

seat [25] UTF8String OPTIONAL,

rd [26] UTF8String OPTIONAL,

rdsec [27] UTF8String OPTIONAL,

rdbr [28] UTF8String OPTIONAL,

rdsubbr [29] UTF8String OPTIONAL,

prm [30] UTF8String OPTIONAL,

pom [31] UTF8String OPTIONAL

}

-- TS 29.572 [24], clause 6.1.6.2.15

PositioningMethodAndUsage ::= SEQUENCE

{

method [1] PositioningMethod,

mode [2] PositioningMode,

usage [3] Usage

}

-- TS 29.572 [24], clause 6.1.6.2.16

GNSSPositioningMethodAndUsage ::= SEQUENCE

{

mode [1] PositioningMode,

gNSS [2] GNSSID,

usage [3] Usage

}

-- TS 29.572 [24], clause 6.1.6.2.6

Point ::= SEQUENCE

{

geographicalCoordinates [1] GeographicalCoordinates

}

-- TS 29.572 [24], clause 6.1.6.2.7

PointUncertaintyCircle ::= SEQUENCE

{

geographicalCoordinates [1] GeographicalCoordinates,

uncertainty [2] Uncertainty

}

-- TS 29.572 [24], clause 6.1.6.2.8

PointUncertaintyEllipse ::= SEQUENCE

{

geographicalCoordinates [1] GeographicalCoordinates,

uncertainty [2] UncertaintyEllipse,

confidence [3] Confidence

}

-- TS 29.572 [24], clause 6.1.6.2.9

Polygon ::= SEQUENCE

{

pointList [1] SET SIZE (3..15) OF GeographicalCoordinates

}

-- TS 29.572 [24], clause 6.1.6.2.10

PointAltitude ::= SEQUENCE

{

point [1] GeographicalCoordinates,

altitude [2] Altitude

}

-- TS 29.572 [24], clause 6.1.6.2.11

PointAltitudeUncertainty ::= SEQUENCE

{

point [1] GeographicalCoordinates,

altitude [2] Altitude,

uncertaintyEllipse [3] UncertaintyEllipse,

uncertaintyAltitude [4] Uncertainty,

confidence [5] Confidence

}

-- TS 29.572 [24], clause 6.1.6.2.12

EllipsoidArc ::= SEQUENCE

{

point [1] GeographicalCoordinates,

innerRadius [2] InnerRadius,

uncertaintyRadius [3] Uncertainty,

offsetAngle [4] Angle,

includedAngle [5] Angle,

confidence [6] Confidence

}

-- TS 29.572 [24], clause 6.1.6.2.4

GeographicalCoordinates ::= SEQUENCE

{

latitude [1] UTF8String,

longitude [2] UTF8String,

mapDatumInformation [3] OGCURN OPTIONAL

}

-- TS 29.572 [24], clause 6.1.6.2.22

UncertaintyEllipse ::= SEQUENCE

{

semiMajor [1] Uncertainty,

semiMinor [2] Uncertainty,

orientationMajor [3] Orientation

}

-- TS 29.572 [24], clause 6.1.6.2.18

HorizontalVelocity ::= SEQUENCE

{

hSpeed [1] HorizontalSpeed,

bearing [2] Angle

}

-- TS 29.572 [24], clause 6.1.6.2.19

HorizontalWithVerticalVelocity ::= SEQUENCE

{

hSpeed [1] HorizontalSpeed,

bearing [2] Angle,

vSpeed [3] VerticalSpeed,

vDirection [4] VerticalDirection

}

-- TS 29.572 [24], clause 6.1.6.2.20

HorizontalVelocityWithUncertainty ::= SEQUENCE

{

hSpeed [1] HorizontalSpeed,

bearing [2] Angle,

uncertainty [3] SpeedUncertainty

}

-- TS 29.572 [24], clause 6.1.6.2.21

HorizontalWithVerticalVelocityAndUncertainty ::= SEQUENCE

{

hspeed [1] HorizontalSpeed,

bearing [2] Angle,

vSpeed [3] VerticalSpeed,

vDirection [4] VerticalDirection,

hUncertainty [5] SpeedUncertainty,

vUncertainty [6] SpeedUncertainty

}

-- The following types are described in TS 29.572 [24], table 6.1.6.3.2-1

Altitude ::= UTF8String

Angle ::= INTEGER (0..360)

Uncertainty ::= INTEGER (0..127)

Orientation ::= INTEGER (0..180)

Confidence ::= INTEGER (0..100)

InnerRadius ::= INTEGER (0..65535)

AgeOfLocationEstimate ::= INTEGER (0..32767)

HorizontalSpeed ::= UTF8String

VerticalSpeed ::= UTF8String

SpeedUncertainty ::= UTF8String

BarometricPressure ::= INTEGER (30000..155000)

-- TS 29.572 [24], clause 6.1.6.3.13

VerticalDirection ::= ENUMERATED

{

upward(1),

downward(2)

}

-- TS 29.572 [24], clause 6.1.6.3.6

PositioningMethod ::= ENUMERATED

{

cellID(1),

eCID(2),

oTDOA(3),

barometricPresure(4),

wLAN(5),

bluetooth(6),

mBS(7),

motionSensor(8)

}

-- TS 29.572 [24], clause 6.1.6.3.7

PositioningMode ::= ENUMERATED

{

uEBased(1),

uEAssisted(2),

conventional(3)

}

-- TS 29.572 [24], clause 6.1.6.3.8

GNSSID ::= ENUMERATED

{

gPS(1),

galileo(2),

sBAS(3),

modernizedGPS(4),

qZSS(5),

gLONASS(6)

}

-- TS 29.572 [24], clause 6.1.6.3.9

Usage ::= ENUMERATED

{

unsuccess(1),

successResultsNotUsed(2),

successResultsUsedToVerifyLocation(3),

successResultsUsedToGenerateLocation(4),

successMethodNotDetermined(5)

}

-- TS 29.571 [17], table 5.2.2-1

TimeZone ::= UTF8String

-- Open Geospatial Consortium URN [35]

OGCURN ::= UTF8String

END

**\*\*\* End of All MODIFICATIONS \*\*\***