**3GPP SA3LI#85e-a *S3i220246***

**eMeeting, 25-29 April 2022**

**Source: OTD**

**Title:** Agreed text for PDN Connection Establishment record

**Document for: Discussion**

**Agenda Item: 8.1**

**Work Item / Release: LI17**

*Abstract of the contribution: This document captures the text that was technically agreed in document draft\_s3i220242r1 but not included in the carried forward into draft\_s3i220242r2. This text includes a parameter addition to the SMFPDUSessionEstablishment record, a change in the description of a parameter from the same record, the details for that parameter from the clause detailing the xIRI Messages for LI at the SMF+PGW-C, updated triggers for use at the SMF+PGW-C and the related ASN.1.*

\*\*\* Start of First Change \*\*\*

##### 6.2.3.2.2 PDU session establishment

The IRI-POI in the SMF, shall generate an xIRI containing an SMFPDUSessionEstablishment record when the IRI-POI present in the SMF detects that a PDU session has been established for the target UE. The IRI-POI present in the SMF shall generate the xIRI for the following events:

- For a non-roaming scenario, the SMF (or for a roaming scenario, V-SMF in the VPLMN), sends the N1 NAS message (via AMF) PDU SESSION ESTABLISHMENT ACCEPT to the UE and the 5G Session Management (5GSM) state within the SMF is changed to PDU SESSION ACTIVE (see TS 24.501 [13]).

- For a home-routed roaming scenario, the SMF in the HPLMN (i.e. H-SMF) sends the N16: Nsmf\_PDU\_Session\_Create response message with n1SmInfoToUe IE containing the PDU SESSION ESTABLISHMENT ACCEPT (see TS 29.502 [16]).

Table 6.2.3-1: Payload for SMFPDUSessionEstablishment record

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| sUPI | SUPI associated with the PDU session (e.g. as provided by the AMF in the associated Nsmf\_PDU\_Session\_CreateSMContext service operation). Shall be present except for PEI-only unauthenticated emergency sessions (see NOTE). | C |
| sUPIUnauthenticated | Shall be present if a SUPI is present in the message and set to “true” if the SUPI has not been authenticated, or “false” if it has been authenticated. | C |
| pEI | PEI associated with the PDU session if available (see NOTE). | C |
| gPSI | GPSI associated with the PDU session if available (see NOTE). | C |
| pDUSessionID | PDU Session ID See TS 24.501 [13] clause 9.4. | M |
| gTPTunnelID | Contains the F-TEID identifying the GTP tunnel used to encapsulate the traffic, as defined in TS 29.244 [15] clause 8.2.3. Non-GTP encapsulation is for further study. | M |
| pDUSessionType | Identifies selected PDU session type, see TS 24.501 [13] clause 9.11.4.11. | M |
| sNSSAI | Slice identifiers associated with the PDU session, if available. See TS 23.003 [19] clause 28.4.2 and TS 23.501 [2] clause 5.15.2. | C |
| uEEndpoint | UE endpoint address(es) if available. | C |
| non3GPPAccessEndpoint | UE's local IP address used to reach the N3IWF, TNGF or TWIF, if available. IP addresses are given as 4 octets (for IPv4) or 16 octets (for IPv6) with the most significant octet first (network byte order). | C |
| location | Location information provided by the AMF, if available.Encoded as a *userLocation* parameter (*location>locationInfo>userLocation*), see Annex A. | C |
| dNN | Data Network Name associated with the target traffic, as defined in TS 23.003[19] clause 9A and described in TS 23.501 [2] clause 4.3.2.2. | M |
| aMFID | Identifier of the AMF associated with the target UE, as defined in TS 23.003 [19] clause 2.10.1 when available. | C |
| hSMFURI | URI of the Nsmf\_PDUSession service of the selected H-SMF, if available. See TS 29.502 [16] clause 6.1.6.2.2. | C |
| requestType | Type of request as described in TS 24.501 [13] clause 9.11.3.47 if available. In the case where the network does not support Multi Access (MA) PDU sessions, but receives a MA PDU session request, a request type of “Initial request” shall be reported. | C |
| accessType | Access type associated with the session (i.e. 3GPP or non-3GPP access) if provided by the AMF (see TS 24.501 [13] clause 9.11.2.1A). | C |
| rATType | RAT Type associated with the access if provided by the AMF as part of session establishment (see TS 23.502 [4] clause 4.3.2). Values given as per TS 29.571 [17] clause 5.4.3.2. | C |
| sMPDUDNRequest | Contents of the SM PDU DN Request container, if available, as described in TS 24.501 [13] clause 9.11.4.15. | C |
| uEEPSPDNConnection | This IE shall be present, if available, during an EPS to 5GS Idle mode mobility or handover using the N26 interface. When present, it shall contain the EPS bearer context(s) information present in the uEEPSPDNConnection parameter of the intercepted SmContextCreateData message. (see TS 29.502 [16] clause 6.1.6.2.2). | C |
| ePS5GSComboInfo | Provides detailed information about PDN Connections associated with PDU Sessions when the SMFPDUSessionEstablishment xIRI message is used to report PDU Session Establishment (See clause 6.3.Cl1.1). Shall be included when the AMF has selected a SMF+PGW-C to serve the PDU session. This parameter shall include the additional IEs in Table 6.2.3-1A, if present. | C |
| ePSPDNConnectionEstablishment | Provides details about PDN Connections when the SMFPDUSessionEstablishment xIRI message is used to report PDN Connection establishment. See Table 6.3.3-Ta6 and clause 6.3.Cl1.1. | C |
| NOTE: At least one of the SUPI, PEI or GPSI fields shall be present. |

Table 6.2.3-1A: Payload for ePS5GSComboInfo

|  |  |  |
| --- | --- | --- |
| ePSInterworkingIndication | Indication that the AMF has selected a SMF+PGW-C to serve the PDU session. See TS 29.502 [16] clause 6.1.6.3.11. | M |
| ePSSubscriberIDs | Includes the Subscriber Identities associated with the EPS PDN Connection in the UE Context sent from the MME to the AMF. See TS 29.274 clause 7.2.1 and TS 23.502 [4] clause 4.11.1.  | M |
| ePSPdnCnxInfo | Indicates that the PDU Session may be moved to EPS During its lifetime. See TS 29.502 [16] clause 6.1.6.2.31. | C |
| ePSBearerInfo | Includes the EPS Bearer context(s) successfully setup in EPS for the PDU Session. See TS 29.502 [16] clause 6.1.6.2.4. | C |

\*\*\* Start of Next Change \*\*\*

### 6.3.Cl1 xIRI Messages for LI at the SMF+PGW-C

#### 6.3.Cl1.1 PDU Session Establishment message reporting PDU session establishment or PDN Connection establishment

The IRI-POI in the SMF+PGW-C shall generate an xIRI containing an SMFPDUSessionEstablishment record (see clause 6.2.3.2.2) when the IRI-POI present in the SMF+PGW-C detects that a PDU Session or PDN Connectoin has been established for the target UE. The IRI-POI present in the SMF+PGW-C shall generate the xIRI for the following events:

- The SMF+PGW-C creates a new PDN Connection in the target UE context of the SMF+PGW-C (see TS 23.401 [50] clause 5.7.4).

- The SMF+PGW-C creates a new PDU Session context or SM Context for the target UE (see TS 29.502 [16] clause 5.2.2.2 and clause 5.2.2.7).

When the SMFPDUSessionEstablishment record (see clause 6.2.3.2.2) is used to report the creation of a new PDN Connection, the ePSPDNConnectionEstablishment field shall be populated with the information in Table 6.3.3-Ta6. If there is no SUPI associated to the SM context for the target UE, the SUPI field of the SMFPDUSessionEstablishment record shall be populated with the value of the IMSI from the target UE context.

Table 6.3.3-Ta6: Payload for ePSPDNConnectionEstablishment field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| ePSSubscriberIDs | EPS Subscriber Identities associated with the PDN connection (e.g. as provided by the MME or SGW in the associated Create Session Request or as associated with the PDN connection in the context known at the NF). The IMSI shall be present except for unauthenticated emergency sessions (see NOTE). | M |
| IMSIUnauthenticated | Shall be present if an IMSI is present in the ePSSubscriberIDs and set to “true” if the IMSI has not been authenticated, or “false” if it has been authenticated. | C |
| defaultBearerID | Shall contain the EPS Bearer Identity of the default bearer associated with the PDN connection. | M |
| gTPTunnelInfo | Contains the information for the Control Plane GTP Tunnels present in the Create Session Request or known in the context at the SGW, PGW or ePDG. See Table 6.3.3-Ta11. | C |
| pDNConnectionType | Identifies selected PDN session type, see TS 29.274 [13] clause 8.34. | M |
| uEEndpoints | UE endpoint address(es) if available. Derived from the PDN Address portion of the PDN Address Allocation parameter (see TS 29.274 [Re1] clause 8.14) present in the Create Session Request or the IP Address associated to the PDN Connection in the context known at the NF (see TS 23.401 clauses 5.7.3 and 5.7.4). | C |
| non3GPPAccessEndpoint | UE's local IP address used to reach the ePDG, if present in the Create Session Request (see TS 29.274 [Re1], clause 7.2.1) or known at the context at the SGW, PGW or ePDG. | C |
| location | Location information present in the Create Session Request (see TS 29.274 [Re1], clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| additionalLocation | Additional location information present in the Create Session Request, known known in the context at the SGW, PGW or ePDG, or known at the MDF. | C |
| aPN | Access Point Name associated with the PDN connection present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.6) or known at the context at the SGW, PGW or ePDG (see TS 23.401 clause 5.6.4), as defined in TS 23.003[19] clause 9.1. | M |
| requestType | Type of request as derived from the Request Type described in TS 24.301 [Re2] clause 9.9.4.14 and TS 24.008 [Re3] clause 10.5.6.17, if available. | C |
| accessType | Access type associated with the PDN connection (i.e. 3GPP or non-3GPP access). Shall be set to nonThreeGPPAccess by the ePDG or by the PGW when the Create Session Request for the PDN connection is received from an ePDG. Shall be set to threeGPPAccess by the SGW or by the PGW when the Create Session Request for the PDN connection is received from an SGW.  | C |
| rATType | RAT Type associated with the PDN connection. Shall be present if included in the Create Session Request (see TS 29.274 [Re1] clause 7.2.1) or known at the context at the SGW, PGW or ePDG (see TS 23.401 clause 5.6.4). | C |
| protocolConfigurationOptions | Shall be present if the Create Session Request or the Create Session Response (see TS 29.274 [Re1] clause 7.2.2 and clause 7.2.3) contains the Protocol Configuration, Additional Protocol Configuration Options or extended Protocol Configuration Options IE. See Table 7.6.3.3-Ta9. | C |
| servingNetwork | Shall be present if this IE is in the Create Session Request or the context for the PDN connection at the SGW/PGW. | C |
| sMPDUDNRequest | Contents of the SM PDU DN Request container, if available, as described in TS 24.501 [13] clause 9.11.4.15. | C |
| bearerContextsCreated | Shall include a list of the Bearer Contexts created sent in the Create Session Response message (see TS 29.274 [Re1] clause 7.2.2). See Table 7.6.3.3-Ta7.  | M |
| bearerContextsMarkedForRemoval | Shall include a list of the Bearer Contexts to be removed sent in the Create Session Response message (see TS 29.274 [Re1] clause 7.2.2). See Table 7.6.3.3-Ta8. | C |
| indicationFlags | Shall be included if the Indication Flags field is present in the Create Session Request (see TS 29.274 [Re1] claus 7.2.1). The value of this parameter shall be set to the value of the Indication IE (see TS 29.274 [Re1] clause 8.12) starting with octet 5. | C |
| handoverIndication | Shall be present if the Handover Indication is set to 1 in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.12). | C |
| nBIFOMSupport | Shall be present if the NBIFOM Support Indication is set to 1 in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.12). | C |
| fiveGSInterworkingInfo | Shall be present if the 5GS Interworking Indication is present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.12). See Table 6.3.3-Ta10. | C |
| cSRMFI | Shall be present if the Create Session Request Message Forwarded Indication (CSRMFI) is present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.12). Indicates the Create Session Request message has been forwarded by a PGW. | C |
| restorationOfPDNConnectionsSupport | Shall be present if the Restoration of PDN connection after an PGW-C/SMF Change Support Indication is present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.12). | C |
| pGWChangeIndication | Shall be present if the PGW Change Indication is present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.12). | C |
| pGWRNSI | Shall be present if the PGW Redirection due to mismatch with Network Slice subscribed by the UE Support Indication is present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.1 and 8.12). | C |

Table 6.3.3-Ta7: Bearer Contexts Created Field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| ePSBearerID | Shall include the EPS bearer ID for the EPS Bearer (See TS 29.274 [Re1] clause 7.2.2). | M |
| cause | Shall indicate whether the bearer handling was successful and if not, it gives information on the reason (see TS 29.274 [Re1] clause 7.2.2). Sent as an integer cause value (see TS 29.274 [Re1] Table 8.4-1)  | M |
| gTPTunnelInfo | Contains the information for the User Plane GTP Tunnels for the bearer context if present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.2 and 8.15) or known at the context at the SGW, PGW or ePDG (see TS 23.401 clause 5.6.4). See Table 6.3.3-Ta11. | C |
| bearerQOS | Shall include the QOS information for the bearer if present in the Create Session Request (see TS 29.274 [Re1] clauses 7.2.2 and 8.15) or known at the context at the SGW, PGW or ePDG (see TS 23.401 clause 5.6.4). See Table 6.3.3-Ta13. | C |

Table 6.3.3-Ta8: Bearer Contexts Marked for Removal Field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| ePSBearerID | Shall include the EPS bearer ID for the EPS Bearer (See TS 29.274 [Re1] clause 7.2.2). | M |
| cause | Shall indicate whether the bearer handling was successful and if not, it gives information on the reason (see TS 29.274 [Re1] clause 7.2.2). | M |

Table 6.3.3-Ta9: Protocol Configuration Options Field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| requestPCO | Shall be present if the Protocol Configuration Options IE is present in the request message. The value of this parameter shall contain a copy of the value field of the PCO IE of the request message (see 29.274 [Re1] clause 8.13 starting with octet 5). | C |
| requestAPCO | Shall be present if the Additional Protocol Configuration Options IE is present in the request message. The value of this parameter shall contain a copy of the value field of the PCO IE of the request message (see 29.274 [Re1] clause 8.94 starting with octet 5). | C |
| requestEPCO | Shall be present if the Extended Protocol Configuration Options IE is present in the request message. The value of this parameter shall contain a copy of the value field of the PCO IE of the request message (see 29.274 [Re1] clause 8.128 starting with octet 5). | C |
| responsePCO | Shall be present if the Protocol Configuration Options IE is present in the response message. The value of this parameter shall contain a copy of the value field of the PCO IE of the response message (see 29.274 [Re1] clause 8.13 starting with octet 5). | C |
| responseAPCO | Shall be present if the Additional Protocol Configuration Options IE is present in the response message. The value of this parameter shall contain a copy of the value field of the PCO IE of the response message (see 29.274 [Re1] clause 8.94 starting with octet 5). | C |
| responseEPCO | Shall be present if the Extended Protocol Configuration Options IE is present in the response message. The value of this parameter shall contain a copy of the value field of the PCO IE of the response message (see 29.274 [Re1] clause 8.128 starting with octet 5). | C |

Table 6.3.3-Ta10: fiveGSInterworkingInfo field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| fiveGSInterworkingIndicator | Shall be set toTRUE if the 5GSIWKI flag in the Indication IE of the Create Session Request is set to 1. Indicates that the UE supports N1 mode and the PDN connection is not restricted from interworking by the 5GS user subscription. See TS 29.274 [Re1] clauses 7.2.1 and 8.12. | M |
| fiveGSInterworkingWithoutN26 | Shall be set to TRUE if the 5GS Interworking without N26 Indication flag in the Indication IE of the Create Session Request is set to 1. If the 5GS Interworking without N26 Indication flag in the Indication IE of the Create Session Request is set to 0 or not present, this parameter shall be set to FALSE. See TS 29.274 [Re1] clauses 7.2.1 and 8.12. | M |
| fiveGCNotRestrictedSupport | Shall be set to True if the 5GCNRS (5GC Not Restricted Support) flag in the Indication IE of the Create Session Request is set to 1. If the 5GCNRS flag in the Indication IE of the Create Session Request is set to 0 or not present, this parameter shall be set to FALSE. See TS 29.274 [Re1] clauses 7.2.1 and 8.12. | M |

Table 6.3.3-Ta11: gTPTunnelInfo field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| ePSGTPTunnels | Shall include the EPS GTP Tunnels (See Table 6.3.3-Ta12). | M |

Table 6.3.3-Ta12: ePSGTPTunnels field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| controlPlaneSenderFTEID | Shall include the Sender F-TEID for the control plane if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| controlPlanePGWS5S8FTEID | Shall include the PGW F-TEID for the control plane if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| s1UeNodeBFTEID | Shall include the F-TEID for the eNodeB S1-U interface for the bearer if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| s5S8SGWFTEID | Shall include the F-TEID for the SGW S5 or S8 interface for the bearer if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| s5S8PGWFTEID | Shall include the F-TEID for the PGW S5 or S8 interface for the bearer if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| s2bUePDGFTEID | Shall include the F-TEID for the ePDG on the S2b-U interface for the bearer if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the PGW or ePDG. | C |
| s2aUePDGFTEID | Shall include the F-TEID for the ePDG on the S2a-U interface for the bearer if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the PGW or ePDG. | C |

Table 6.3.3-Ta13: bearerQOS Field

|  |  |  |
| --- | --- | --- |
| Field name | Description | M/C/O |
| qCI | Shall include the QCI for the bearer if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1), the Create Session Response (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| maximumUplinkBitRate | Shall include the maximum uplink bitrate encoded as kilobits per second in binary value (see TS 29.274 [Re1] clause 8.15) if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1), the Create Session Response (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| maximumDownlinkBitRate | Shall include the maximum downlink bitrate encoded as kilobits per second in binary value (see TS 29.274 [Re1] clause 8.15) if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1), the Create Session Response (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| guaranteedUplinkBitRate | Shall include the guaranteed uplink bitrate encoded as kilobits per second in binary value (see TS 29.274 [Re1] clause 8.15) if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1), the Create Session Response (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| guaranteedDownlinkBitRate | Shall include the guaranteed downlink bitrate encoded as kilobits per second in binary value (see TS 29.274 [Re1] clause 8.15) if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1), the Create Session Response (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |
| priorityLevel | Shall include the priority level assigned to the bearer as an integer value (see TS 29.274 [Re1] clause 8.15) if present in the Create Session Request (See TS 29.274 [Re1] clause 7.2.1), the Create Session Response (See TS 29.274 [Re1] clause 7.2.1) or known in the context at the SGW, PGW or ePDG. | C |

\*\*\* Start of Next Change \*\*\*

aaTS33128Payloads

{itu-t(0) identified-organization(4) etsi(0) securityDomain(2) lawfulIntercept(2) threeGPP(4) ts33128(19) r17(17) version3(3)}

DEFINITIONS IMPLICIT TAGS EXTENSIBILITY IMPLIED ::=

BEGIN

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

-- Relative OIDs

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

tS33128PayloadsOID RELATIVE-OID ::= {threeGPP(4) ts33128(19) r17(17) version3(3)}

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, see also sMSReport ([56] below)

 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,

 -- More Subscriber-management related events, see clause 7.2.2

 subscriberRecordChangeMessage [54] UDMSubscriberRecordChangeMessage,

 cancelLocationMessage [55] UDMCancelLocationMessage,

 -- SMS-related events continued from choice 12

 sMSReport [56] SMSReport,

 -- MA PDU session-related events, see clause 6.2.3.2.7

 sMFMAPDUSessionEstablishment [57] SMFMAPDUSessionEstablishment,

 sMFMAPDUSessionModification [58] SMFMAPDUSessionModification,

 sMFMAPDUSessionRelease [59] SMFMAPDUSessionRelease,

 startOfInterceptionWithEstablishedMAPDUSession [60] SMFStartOfInterceptionWithEstablishedMAPDUSession,

 unsuccessfulMASMProcedure [61] SMFMAUnsuccessfulProcedure,

 -- Identifier Association events, see clauses 6.2.2.2.7 and 6.3.2.2.2

 aMFIdentifierAssociation [62] AMFIdentifierAssociation,

 mMEIdentifierAssociation [63] MMEIdentifierAssociation,

 -- PDU to MA PDU session-related events, see clause 6.2.3.2.8

 sMFPDUtoMAPDUSessionModification [64] SMFPDUtoMAPDUSessionModification,

 -- NEF services related events, see clause 7.7.2

 nEFPDUSessionEstablishment [65] NEFPDUSessionEstablishment,

 nEFPDUSessionModification [66] NEFPDUSessionModification,

 nEFPDUSessionRelease [67] NEFPDUSessionRelease,

 nEFUnsuccessfulProcedure [68] NEFUnsuccessfulProcedure,

 nEFStartOfInterceptionWithEstablishedPDUSession [69] NEFStartOfInterceptionWithEstablishedPDUSession,

 nEFdeviceTrigger [70] NEFDeviceTrigger,

 nEFdeviceTriggerReplace [71] NEFDeviceTriggerReplace,

 nEFdeviceTriggerCancellation [72] NEFDeviceTriggerCancellation,

 nEFdeviceTriggerReportNotify [73] NEFDeviceTriggerReportNotify,

 nEFMSISDNLessMOSMS [74] NEFMSISDNLessMOSMS,

 nEFExpectedUEBehaviourUpdate [75] NEFExpectedUEBehaviourUpdate,

 -- SCEF services related events, see clause 7.8.2

 sCEFPDNConnectionEstablishment [76] SCEFPDNConnectionEstablishment,

 sCEFPDNConnectionUpdate [77] SCEFPDNConnectionUpdate,

 sCEFPDNConnectionRelease [78] SCEFPDNConnectionRelease,

 sCEFUnsuccessfulProcedure [79] SCEFUnsuccessfulProcedure,

 sCEFStartOfInterceptionWithEstablishedPDNConnection [80] SCEFStartOfInterceptionWithEstablishedPDNConnection,

 sCEFdeviceTrigger [81] SCEFDeviceTrigger,

 sCEFdeviceTriggerReplace [82] SCEFDeviceTriggerReplace,

 sCEFdeviceTriggerCancellation [83] SCEFDeviceTriggerCancellation,

 sCEFdeviceTriggerReportNotify [84] SCEFDeviceTriggerReportNotify,

 sCEFMSISDNLessMOSMS [85] SCEFMSISDNLessMOSMS,

 sCEFCommunicationPatternUpdate [86] SCEFCommunicationPatternUpdate,

 -- EPS Events, see clause 6.3

 -- MME Events, see clause 6.3.2.2

 mMEAttach [87] MMEAttach,

 mMEDetach [88] MMEDetach,

 mMELocationUpdate [89] MMELocationUpdate,

 mMEStartOfInterceptionWithEPSAttachedUE [90] MMEStartOfInterceptionWithEPSAttachedUE,

 mMEUnsuccessfulProcedure [91] MMEUnsuccessfulProcedure,

 -- AKMA key management events, see clause 7.9.1

 aAnFAnchorKeyRegister [92] AAnFAnchorKeyRegister,

 aAnFKAKMAApplicationKeyGet [93] AAnFKAKMAApplicationKeyGet,

 aAnFStartOfInterceptWithEstablishedAKMAKeyMaterial [94] AAnFStartOfInterceptWithEstablishedAKMAKeyMaterial,

 aAnFAKMAContextRemovalRecord [95] AAnFAKMAContextRemovalRecord,

 aFAKMAApplicationKeyRefresh [96] AFAKMAApplicationKeyRefresh,

 aFStartOfInterceptWithEstablishedAKMAApplicationKey [97] AFStartOfInterceptWithEstablishedAKMAApplicationKey,

 aFAuxiliarySecurityParameterEstablishment [98] AFAuxiliarySecurityParameterEstablishment,

 aFApplicationKeyRemoval [99] AFApplicationKeyRemoval,

 -- HR LI Events, see clause 7.10.3.3

 n9HRPDUSessionInfo [100] N9HRPDUSessionInfo,

 s8HRBearerInfo [101] S8HRBearerInfo,

 -- Separated Location Reporting, see clause 7.3.4

 separatedLocationReporting [102] SeparatedLocationReporting,

 -- STIR SHAKEN and RCD/eCNAM Events, see clause 7.11.2

 sTIRSHAKENSignatureGeneration [103] STIRSHAKENSignatureGeneration,

 sTIRSHAKENSignatureValidation [104] STIRSHAKENSignatureValidation,

 -- IMS events, see clause 7.11.4.2

 iMSMessage [105] IMSMessage,

 startOfInterceptionForActiveIMSSession [106] StartOfInterceptionForActiveIMSSession,

 iMSCCUnavailable [107] IMSCCUnavailable,

 -- UDM events, see clause 7.2.2

 uDMLocationInformationResult [108] UDMLocationInformationResult,

 uDMUEInformationResponse [109] UDMUEInformationResponse,

 uDMUEAuthenticationResponse [110] UDMUEAuthenticationResponse,

 -- AMF events, see 6.2.2.2.8

 positioningInfoTransfer [111] AMFPositioningInfoTransfer,

 -- MME Events, see clause 6.3.2.2.8

 mMEPositioningInfoTransfer [112] MMEPositioningInfoTransfer

}

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

-- 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, see also sMSReport ([56] below)

 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 7.3.2

 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,

 -- More Subscriber-management related events, see clause 7.2.2

 subscriberRecordChangeMessage [54] UDMSubscriberRecordChangeMessage,

 cancelLocationMessage [55] UDMCancelLocationMessage,

 -- SMS-related events, continued from choice 12

 sMSReport [56] SMSReport,

 -- MA PDU session-related events, see clause 6.2.3.2.7

 sMFMAPDUSessionEstablishment [57] SMFMAPDUSessionEstablishment,

 sMFMAPDUSessionModification [58] SMFMAPDUSessionModification,

 sMFMAPDUSessionRelease [59] SMFMAPDUSessionRelease,

 startOfInterceptionWithEstablishedMAPDUSession [60] SMFStartOfInterceptionWithEstablishedMAPDUSession,

 unsuccessfulMASMProcedure [61] SMFMAUnsuccessfulProcedure,

 -- Identifier Association events, see clauses 6.2.2.2.7 and 6.3.2.2.2

 aMFIdentifierAssociation [62] AMFIdentifierAssociation,

 mMEIdentifierAssociation [63] MMEIdentifierAssociation,

 -- PDU to MA PDU session-related events, see clause 6.2.3.2.8

 sMFPDUtoMAPDUSessionModification [64] SMFPDUtoMAPDUSessionModification,

 -- NEF services related events, see clause 7.7.2,

 nEFPDUSessionEstablishment [65] NEFPDUSessionEstablishment,

 nEFPDUSessionModification [66] NEFPDUSessionModification,

 nEFPDUSessionRelease [67] NEFPDUSessionRelease,

 nEFUnsuccessfulProcedure [68] NEFUnsuccessfulProcedure,

 nEFStartOfInterceptionWithEstablishedPDUSession [69] NEFStartOfInterceptionWithEstablishedPDUSession,

 nEFdeviceTrigger [70] NEFDeviceTrigger,

 nEFdeviceTriggerReplace [71] NEFDeviceTriggerReplace,

 nEFdeviceTriggerCancellation [72] NEFDeviceTriggerCancellation,

 nEFdeviceTriggerReportNotify [73] NEFDeviceTriggerReportNotify,

 nEFMSISDNLessMOSMS [74] NEFMSISDNLessMOSMS,

 nEFExpectedUEBehaviourUpdate [75] NEFExpectedUEBehaviourUpdate,

 -- SCEF services related events, see clause 7.8.2

 sCEFPDNConnectionEstablishment [76] SCEFPDNConnectionEstablishment,

 sCEFPDNConnectionUpdate [77] SCEFPDNConnectionUpdate,

 sCEFPDNConnectionRelease [78] SCEFPDNConnectionRelease,

 sCEFUnsuccessfulProcedure [79] SCEFUnsuccessfulProcedure,

 sCEFStartOfInterceptionWithEstablishedPDNConnection [80] SCEFStartOfInterceptionWithEstablishedPDNConnection,

 sCEFdeviceTrigger [81] SCEFDeviceTrigger,

 sCEFdeviceTriggerReplace [82] SCEFDeviceTriggerReplace,

 sCEFdeviceTriggerCancellation [83] SCEFDeviceTriggerCancellation,

 sCEFdeviceTriggerReportNotify [84] SCEFDeviceTriggerReportNotify,

 sCEFMSISDNLessMOSMS [85] SCEFMSISDNLessMOSMS,

 sCEFCommunicationPatternUpdate [86] SCEFCommunicationPatternUpdate,

 -- EPS Events, see clause 6.3

 -- MME Events, see clause 6.3.2.2

 mMEAttach [87] MMEAttach,

 mMEDetach [88] MMEDetach,

 mMELocationUpdate [89] MMELocationUpdate,

 mMEStartOfInterceptionWithEPSAttachedUE [90] MMEStartOfInterceptionWithEPSAttachedUE,

 mMEUnsuccessfulProcedure [91] MMEUnsuccessfulProcedure,

 -- AKMA key management events, see clause 7.9.1

 aAnFAnchorKeyRegister [92] AAnFAnchorKeyRegister,

 aAnFKAKMAApplicationKeyGet [93] AAnFKAKMAApplicationKeyGet,

 aAnFStartOfInterceptWithEstablishedAKMAKeyMaterial [94] AAnFStartOfInterceptWithEstablishedAKMAKeyMaterial,

 aAnFAKMAContextRemovalRecord [95] AAnFAKMAContextRemovalRecord,

 aFAKMAApplicationKeyRefresh [96] AFAKMAApplicationKeyRefresh,

 aFStartOfInterceptWithEstablishedAKMAApplicationKey [97] AFStartOfInterceptWithEstablishedAKMAApplicationKey,

 aFAuxiliarySecurityParameterEstablishment [98] AFAuxiliarySecurityParameterEstablishment,

 aFApplicationKeyRemoval [99] AFApplicationKeyRemoval,

 -- tag 100 is reserved because there is no equivalent n9HRPDUSessionInfo in IRIEvent.

 -- tag 101 is reserved because there is no equivalent S8HRBearerInfo in IRIEvent.

 -- Separated Location Reporting, see clause 7.3.4

 separatedLocationReporting [102] SeparatedLocationReporting,

 -- STIR SHAKEN and RCD/eCNAM Events, see clause 7.11.3

 sTIRSHAKENSignatureGeneration [103] STIRSHAKENSignatureGeneration,

 sTIRSHAKENSignatureValidation [104] STIRSHAKENSignatureValidation,

 -- IMS events, see clause 7.11.4.2

 iMSMessage [105] IMSMessage,

 startOfInterceptionForActiveIMSSession [106] StartOfInterceptionForActiveIMSSession,

 iMSCCUnavailable [107] IMSCCUnavailable,

 -- UDM events, see clause 7.2.2

 uDMLocationInformationResultRecord [108] UDMLocationInformationResult,

 uDMUEInformationResponse [109] UDMUEInformationResponse,

 uDMUEAuthenticationResponse [110] UDMUEAuthenticationResponse,

 -- AMF events, see 6.2.2.2.8

 positioningInfoTransfer [111] AMFPositioningInfoTransfer,

 -- MME Events, see clause 6.3.2.2.8

 mMEPositioningInfoTransfer [112] MMEPositioningInfoTransfer

}

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,

 nIDDCCPDU [4] NIDDCCPDU,

 pTCCCPDU [5] PTCCCPDU

}

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

-- HI4 LI notification payload

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

LINotificationPayload ::= SEQUENCE

{

 lINotificationPayloadOID [1] RELATIVE-OID,

 notification [2] LINotificationMessage

}

LINotificationMessage ::= CHOICE

{

 lINotification [1] LINotification

}

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

-- HR LI definitions

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

N9HRPDUSessionInfo ::= SEQUENCE

{

 sUPI [1] SUPI,

 pEI [2] PEI OPTIONAL,

 pDUSessionID [3] PDUSessionID,

 location [4] Location OPTIONAL,

 sNSSAI [5] SNSSAI OPTIONAL,

 dNN [6] DNN OPTIONAL,

 messageCause [7] N9HRMessageCause

}

S8HRBearerInfo ::= SEQUENCE

{

 iMSI [1] IMSI,

 iMEI [2] IMEI OPTIONAL,

 bearerID [3] EPSBearerID,

 linkedBearerID [4] EPSBearerID OPTIONAL,

 location [5] Location OPTIONAL,

 aPN [6] APN OPTIONAL,

 sGWIPAddress [7] IPAddress OPTIONAL,

 messageCause [8] S8HRMessageCause

}

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

-- HR LI parameters

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

N9HRMessageCause ::= ENUMERATED

{

 pDUSessionEstablished(1),

 pDUSessionModified(2),

 pDUSessionReleased(3),

 updatedLocationAvailable(4),

 sMFChanged(5),

 other(6),

 hRLIEnabled(7)

}

S8HRMessageCause ::= ENUMERATED

{

 bearerActivated(1),

 bearerModified(2),

 bearerDeleted(3),

 pDNDisconnected(4),

 updatedLocationAvailable(5),

 sGWChanged(6),

 other(7),

 hRLIEnabled(8)

}

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

-- 5G NEF definitions

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

-- See clause 7.7.2.1.2 for details of this structure

NEFPDUSessionEstablishment ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 pDUSessionID [3] PDUSessionID,

 sNSSAI [4] SNSSAI,

 nEFID [5] NEFID,

 dNN [6] DNN,

 rDSSupport [7] RDSSupport,

 sMFID [8] SMFID,

 aFID [9] AFID

}

-- See clause 7.7.2.1.3 for details of this structure

NEFPDUSessionModification ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 sNSSAI [3] SNSSAI,

 initiator [4] Initiator,

 rDSSourcePortNumber [5] RDSPortNumber OPTIONAL,

 rDSDestinationPortNumber [6] RDSPortNumber OPTIONAL,

 applicationID [7] ApplicationID OPTIONAL,

 aFID [8] AFID OPTIONAL,

 rDSAction [9] RDSAction OPTIONAL,

 serializationFormat [10] SerializationFormat OPTIONAL

}

-- See clause 7.7.2.1.4 for details of this structure

NEFPDUSessionRelease ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 pDUSessionID [3] PDUSessionID,

 timeOfFirstPacket [4] Timestamp OPTIONAL,

 timeOfLastPacket [5] Timestamp OPTIONAL,

 uplinkVolume [6] INTEGER OPTIONAL,

 downlinkVolume [7] INTEGER OPTIONAL,

 releaseCause [8] NEFReleaseCause

}

-- See clause 7.7.2.1.5 for details of this structure

NEFUnsuccessfulProcedure ::= SEQUENCE

{

 failureCause [1] NEFFailureCause,

 sUPI [2] SUPI,

 gPSI [3] GPSI OPTIONAL,

 pDUSessionID [4] PDUSessionID,

 dNN [5] DNN OPTIONAL,

 sNSSAI [6] SNSSAI OPTIONAL,

 rDSDestinationPortNumber [7] RDSPortNumber,

 applicationID [8] ApplicationID,

 aFID [9] AFID

}

-- See clause 7.7.2.1.6 for details of this structure

NEFStartOfInterceptionWithEstablishedPDUSession ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 pDUSessionID [3] PDUSessionID,

 dNN [4] DNN,

 sNSSAI [5] SNSSAI,

 nEFID [6] NEFID,

 rDSSupport [7] RDSSupport,

 sMFID [8] SMFID,

 aFID [9] AFID

}

-- See clause 7.7.3.1.1 for details of this structure

NEFDeviceTrigger ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 triggerId [3] TriggerID,

 aFID [4] AFID,

 triggerPayload [5] TriggerPayload OPTIONAL,

 validityPeriod [6] INTEGER OPTIONAL,

 priorityDT [7] PriorityDT OPTIONAL,

 sourcePortId [8] PortNumber OPTIONAL,

 destinationPortId [9] PortNumber OPTIONAL

}

-- See clause 7.7.3.1.2 for details of this structure

NEFDeviceTriggerReplace ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 triggerId [3] TriggerID,

 aFID [4] AFID,

 triggerPayload [5] TriggerPayload OPTIONAL,

 validityPeriod [6] INTEGER OPTIONAL,

 priorityDT [7] PriorityDT OPTIONAL,

 sourcePortId [8] PortNumber OPTIONAL,

 destinationPortId [9] PortNumber OPTIONAL

}

-- See clause 7.7.3.1.3 for details of this structure

NEFDeviceTriggerCancellation ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 triggerId [3] TriggerID

}

-- See clause 7.7.3.1.4 for details of this structure

NEFDeviceTriggerReportNotify ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 triggerId [3] TriggerID,

 deviceTriggerDeliveryResult [4] DeviceTriggerDeliveryResult

}

-- See clause 7.7.4.1.1 for details of this structure

NEFMSISDNLessMOSMS ::= SEQUENCE

{

 sUPI [1] SUPI,

 gPSI [2] GPSI,

 terminatingSMSParty [3] AFID,

 sMS [4] SMSTPDUData OPTIONAL,

 sourcePort [5] PortNumber OPTIONAL,

 destinationPort [6] PortNumber OPTIONAL

}

-- See clause 7.7.5.1.1 for details of this structure

NEFExpectedUEBehaviourUpdate ::= SEQUENCE

{

 gPSI [1] GPSI,

 expectedUEMovingTrajectory [2] SEQUENCE OF UMTLocationArea5G OPTIONAL,

 stationaryIndication [3] StationaryIndication OPTIONAL,

 communicationDurationTime [4] INTEGER OPTIONAL,

 periodicTime [5] INTEGER OPTIONAL,

 scheduledCommunicationTime [6] ScheduledCommunicationTime OPTIONAL,

 scheduledCommunicationType [7] ScheduledCommunicationType OPTIONAL,

 batteryIndication [8] BatteryIndication OPTIONAL,

 trafficProfile [9] TrafficProfile OPTIONAL,

 expectedTimeAndDayOfWeekInTrajectory [10] SEQUENCE OF UMTLocationArea5G OPTIONAL,

 aFID [11] AFID,

 validityTime [12] Timestamp OPTIONAL

}

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

-- Common SCEF/NEF parameters

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

RDSSupport ::= BOOLEAN

RDSPortNumber ::= INTEGER (0..15)

RDSAction ::= ENUMERATED

{

 reservePort(1),

 releasePort(2)

}

SerializationFormat ::= ENUMERATED

{

 xml(1),

 json(2),

 cbor(3)

}

ApplicationID ::= OCTET STRING

NIDDCCPDU ::= OCTET STRING

TriggerID ::= UTF8String

PriorityDT ::= ENUMERATED

{

 noPriority(1),

 priority(2)

}

TriggerPayload ::= OCTET STRING

DeviceTriggerDeliveryResult ::= ENUMERATED

{

 success(1),

 unknown(2),

 failure(3),

 triggered(4),

 expired(5),

 unconfirmed(6),

 replaced(7),

 terminate(8)

}

StationaryIndication ::= ENUMERATED

{

 stationary(1),

 mobile(2)

}

BatteryIndication ::= ENUMERATED

{

 batteryRecharge(1),

 batteryReplace(2),

 batteryNoRecharge(3),

 batteryNoReplace(4),

 noBattery(5)

}

ScheduledCommunicationTime ::= SEQUENCE

{

 days [1] SEQUENCE OF Daytime

}

UMTLocationArea5G ::= SEQUENCE

{

 timeOfDay [1] Daytime,

 durationSec [2] INTEGER,

 location [3] NRLocation

}

Daytime ::= SEQUENCE

{

 daysOfWeek [1] Day OPTIONAL,

 timeOfDayStart [2] Timestamp OPTIONAL,

 timeOfDayEnd [3] Timestamp OPTIONAL

}

Day ::= ENUMERATED

{

 monday(1),

 tuesday(2),

 wednesday(3),

 thursday(4),

 friday(5),

 saturday(6),

 sunday(7)

}

TrafficProfile ::= ENUMERATED

{

 singleTransUL(1),

 singleTransDL(2),

 dualTransULFirst(3),

 dualTransDLFirst(4),

 multiTrans(5)

}

ScheduledCommunicationType ::= ENUMERATED

{

 downlinkOnly(1),

 uplinkOnly(2),

 bidirectional(3)

}

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

-- 5G NEF parameters

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

NEFFailureCause ::= ENUMERATED

{

 userUnknown(1),

 niddConfigurationNotAvailable(2),

 contextNotFound(3),

 portNotFree(4),

 portNotAssociatedWithSpecifiedApplication(5)

}

NEFReleaseCause ::= ENUMERATED

{

 sMFRelease(1),

 dNRelease(2),

 uDMRelease(3),

 cHFRelease(4),

 localConfigurationPolicy(5),

 unknownCause(6)

}

AFID ::= UTF8String

NEFID ::= UTF8String

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

-- SCEF definitions

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

-- See clause 7.8.2.1.2 for details of this structure

SCEFPDNConnectionEstablishment ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifier [3] NAI OPTIONAL,

 iMEI [4] IMEI OPTIONAL,

 ePSBearerID [5] EPSBearerID,

 sCEFID [6] SCEFID,

 aPN [7] APN,

 rDSSupport [8] RDSSupport,

 sCSASID [9] SCSASID

}

-- See clause 7.8.2.1.3 for details of this structure

SCEFPDNConnectionUpdate ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifier [3] NAI OPTIONAL,

 initiator [4] Initiator,

 rDSSourcePortNumber [5] RDSPortNumber OPTIONAL,

 rDSDestinationPortNumber [6] RDSPortNumber OPTIONAL,

 applicationID [7] ApplicationID OPTIONAL,

 sCSASID [8] SCSASID OPTIONAL,

 rDSAction [9] RDSAction OPTIONAL,

 serializationFormat [10] SerializationFormat OPTIONAL

}

-- See clause 7.8.2.1.4 for details of this structure

SCEFPDNConnectionRelease ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifier [3] NAI OPTIONAL,

 ePSBearerID [4] EPSBearerID,

 timeOfFirstPacket [5] Timestamp OPTIONAL,

 timeOfLastPacket [6] Timestamp OPTIONAL,

 uplinkVolume [7] INTEGER OPTIONAL,

 downlinkVolume [8] INTEGER OPTIONAL,

 releaseCause [9] SCEFReleaseCause

}

-- See clause 7.8.2.1.5 for details of this structure

SCEFUnsuccessfulProcedure ::= SEQUENCE

{

 failureCause [1] SCEFFailureCause,

 iMSI [2] IMSI OPTIONAL,

 mSISDN [3] MSISDN OPTIONAL,

 externalIdentifier [4] NAI OPTIONAL,

 ePSBearerID [5] EPSBearerID,

 aPN [6] APN,

 rDSDestinationPortNumber [7] RDSPortNumber OPTIONAL,

 applicationID [8] ApplicationID OPTIONAL,

 sCSASID [9] SCSASID

}

-- See clause 7.8.2.1.6 for details of this structure

SCEFStartOfInterceptionWithEstablishedPDNConnection ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifier [3] NAI OPTIONAL,

 iMEI [4] IMEI OPTIONAL,

 ePSBearerID [5] EPSBearerID,

 sCEFID [6] SCEFID,

 aPN [7] APN,

 rDSSupport [8] RDSSupport,

 sCSASID [9] SCSASID

}

-- See clause 7.8.3.1.1 for details of this structure

SCEFDeviceTrigger ::= SEQUENCE

{

 iMSI [1] IMSI,

 mSISDN [2] MSISDN,

 externalIdentifier [3] NAI,

 triggerId [4] TriggerID,

 sCSASID [5] SCSASID OPTIONAL,

 triggerPayload [6] TriggerPayload OPTIONAL,

 validityPeriod [7] INTEGER OPTIONAL,

 priorityDT [8] PriorityDT OPTIONAL,

 sourcePortId [9] PortNumber OPTIONAL,

 destinationPortId [10] PortNumber OPTIONAL

}

-- See clause 7.8.3.1.2 for details of this structure

SCEFDeviceTriggerReplace ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifier [3] NAI OPTIONAL,

 triggerId [4] TriggerID,

 sCSASID [5] SCSASID OPTIONAL,

 triggerPayload [6] TriggerPayload OPTIONAL,

 validityPeriod [7] INTEGER OPTIONAL,

 priorityDT [8] PriorityDT OPTIONAL,

 sourcePortId [9] PortNumber OPTIONAL,

 destinationPortId [10] PortNumber OPTIONAL

}

-- See clause 7.8.3.1.3 for details of this structure

SCEFDeviceTriggerCancellation ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifier [3] NAI OPTIONAL,

 triggerId [4] TriggerID

}

-- See clause 7.8.3.1.4 for details of this structure

SCEFDeviceTriggerReportNotify ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifier [3] NAI OPTIONAL,

 triggerId [4] TriggerID,

 deviceTriggerDeliveryResult [5] DeviceTriggerDeliveryResult

}

-- See clause 7.8.4.1.1 for details of this structure

SCEFMSISDNLessMOSMS ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 externalIdentifie [3] NAI OPTIONAL,

 terminatingSMSParty [4] SCSASID,

 sMS [5] SMSTPDUData OPTIONAL,

 sourcePort [6] PortNumber OPTIONAL,

 destinationPort [7] PortNumber OPTIONAL

}

-- See clause 7.8.5.1.1 for details of this structure

SCEFCommunicationPatternUpdate ::= SEQUENCE

{

 mSISDN [1] MSISDN OPTIONAL,

 externalIdentifier [2] NAI OPTIONAL,

 periodicCommunicationIndicator [3] PeriodicCommunicationIndicator OPTIONAL,

 communicationDurationTime [4] INTEGER OPTIONAL,

 periodicTime [5] INTEGER OPTIONAL,

 scheduledCommunicationTime [6] ScheduledCommunicationTime OPTIONAL,

 scheduledCommunicationType [7] ScheduledCommunicationType OPTIONAL,

 stationaryIndication [8] StationaryIndication OPTIONAL,

 batteryIndication [9] BatteryIndication OPTIONAL,

 trafficProfile [10] TrafficProfile OPTIONAL,

 expectedUEMovingTrajectory [11] SEQUENCE OF UMTLocationArea5G OPTIONAL,

 sCSASID [13] SCSASID,

 validityTime [14] Timestamp OPTIONAL

}

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

-- SCEF parameters

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

SCEFFailureCause ::= ENUMERATED

{

 userUnknown(1),

 niddConfigurationNotAvailable(2),

 invalidEPSBearer(3),

 operationNotAllowed(4),

 portNotFree(5),

 portNotAssociatedWithSpecifiedApplication(6)

}

SCEFReleaseCause ::= ENUMERATED

{

 mMERelease(1),

 dNRelease(2),

 hSSRelease(3),

 localConfigurationPolicy(4),

 unknownCause(5)

}

SCSASID ::= UTF8String

SCEFID ::= UTF8String

PeriodicCommunicationIndicator ::= ENUMERATED

{

 periodic(1),

 nonPeriodic(2)

}

EPSBearerID ::= INTEGER (0..255)

APN ::= UTF8String

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

-- AKMA AAnF definitions

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

AAnFAnchorKeyRegister ::= SEQUENCE

{

 aKID [1] NAI,

 sUPI [2] SUPI,

 kAKMA [3] KAKMA OPTIONAL

}

AAnFKAKMAApplicationKeyGet ::= SEQUENCE

{

 type [1] KeyGetType,

 aKID [2] NAI,

 keyInfo [3] AFKeyInfo

}

AAnFStartOfInterceptWithEstablishedAKMAKeyMaterial ::= SEQUENCE

{

 aKID [1] NAI,

 kAKMA [2] KAKMA OPTIONAL,

 aFKeyList [3] SEQUENCE OF AFKeyInfo OPTIONAL

}

AAnFAKMAContextRemovalRecord ::= SEQUENCE

{

 aKID [1] NAI,

 nFID [2] NFID

}

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

-- AKMA common parameters

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

FQDN ::= UTF8String

NFID ::= UTF8String

UAProtocolID ::= OCTET STRING (SIZE(5))

AKMAAFID ::= SEQUENCE

{

 aFFQDN [1] FQDN,

 uaProtocolID [2] UAProtocolID

}

UAStarParams ::= CHOICE

{

 tls12 [1] TLS12UAStarParams,

 generic [2] GenericUAStarParams

}

GenericUAStarParams ::= SEQUENCE

{

 genericClientParams [1] OCTET STRING,

 genericServerParams [2] OCTET STRING

}

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

-- Specific UaStarParmas for TLS 1.2 (RFC5246)

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

TLSCipherType ::= ENUMERATED

{

 stream(1),

 block(2),

 aead(3)

}

TLSCompressionAlgorithm ::= ENUMERATED

{

 null(1),

 deflate(2)

}

TLSPRFAlgorithm ::= ENUMERATED

{

 rfc5246(1)

}

TLSCipherSuite ::= SEQUENCE (SIZE(2)) OF INTEGER (0..255)

TLS12UAStarParams ::= SEQUENCE

{

 preMasterSecret [1] OCTET STRING (SIZE(6)) OPTIONAL,

 masterSecret [2] OCTET STRING (SIZE(6)),

 pRFAlgorithm [3] TLSPRFAlgorithm,

 cipherSuite [4] TLSCipherSuite,

 cipherType [5] TLSCipherType,

 encKeyLength [6] INTEGER (0..255),

 blockLength [7] INTEGER (0..255),

 fixedIVLength [8] INTEGER (0..255),

 recordIVLength [9] INTEGER (0..255),

 macLength [10] INTEGER (0..255),

 macKeyLength [11] INTEGER (0..255),

 compressionAlgorithm [12] TLSCompressionAlgorithm,

 clientRandom [13] OCTET STRING (SIZE(4)),

 serverRandom [14] OCTET STRING (SIZE(4)),

 clientSequenceNumber [15] INTEGER,

 serverSequenceNumber [16] INTEGER,

 sessionID [17] OCTET STRING (SIZE(0..32)),

 tLSExtensions [18] OCTET STRING (SIZE(0..65535))

}

KAF ::= OCTET STRING

KAKMA ::= OCTET STRING

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

-- AKMA AAnF parameters

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

KeyGetType ::= ENUMERATED

{

 internal(1),

 external(2)

}

AFKeyInfo ::= SEQUENCE

{

 aFID [1] AKMAAFID,

 kAF [2] KAF,

 kAFExpTime [3] KAFExpiryTime

}

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

-- AKMA AF definitions

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

AFAKMAApplicationKeyRefresh ::= SEQUENCE

{

 aFID [1] AFID,

 aKID [2] NAI,

 kAF [3] KAF,

 uaStarParams [4] UAStarParams OPTIONAL

}

AFStartOfInterceptWithEstablishedAKMAApplicationKey ::= SEQUENCE

{

 aFID [1] FQDN,

 aKID [2] NAI,

 kAFParamList [3] SEQUENCE OF AFSecurityParams

}

AFAuxiliarySecurityParameterEstablishment ::= SEQUENCE

{

 aFSecurityParams [1] AFSecurityParams

}

AFSecurityParams ::= SEQUENCE

{

 aFID [1] AFID,

 aKID [2] NAI,

 kAF [3] KAF,

 uaStarParams [4] UAStarParams

}

AFApplicationKeyRemoval ::= SEQUENCE

{

 aFID [1] AFID,

 aKID [2] NAI,

 removalCause [3] AFKeyRemovalCause

}

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

-- AKMA AF parameters

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

KAFParams ::= SEQUENCE

{

 aKID [1] NAI,

 kAF [2] KAF,

 kAFExpTime [3] KAFExpiryTime,

 uaStarParams [4] UAStarParams

}

KAFExpiryTime ::= GeneralizedTime

AFKeyRemovalCause ::= ENUMERATED

{

 unknown(1),

 keyExpiry(2),

 applicationSpecific(3)

}

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

-- 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,

 fiveGSTAIList [11] TAIList OPTIONAL,

 sMSOverNasIndicator [12] SMSOverNASIndicator OPTIONAL,

 oldGUTI [13] EPS5GGUTI OPTIONAL,

 eMM5GRegStatus [14] EMM5GMMStatus OPTIONAL,

 nonIMEISVPEI [15] NonIMEISVPEI OPTIONAL,

 mACRestIndicator [16] MACRestrictionIndicator 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,

 switchOffIndicator [10] SwitchOffIndicator OPTIONAL,

 reRegRequiredIndicator [11] ReRegRequiredIndicator 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,

 sMSOverNASIndicator [7] SMSOverNASIndicator OPTIONAL,

 oldGUTI [8] EPS5GGUTI OPTIONAL

}

-- 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,

 fiveGSTAIList [12] TAIList OPTIONAL,

 sMSOverNASIndicator [13] SMSOverNASIndicator OPTIONAL,

 oldGUTI [14] EPS5GGUTI OPTIONAL,

 eMM5GRegStatus [15] EMM5GMMStatus 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

}

-- See clause 6.2.2.2.8 on for details of this structure

AMFPositioningInfoTransfer ::= SEQUENCE

{

 sUPI [1] SUPI,

 sUCI [2] SUCI OPTIONAL,

 pEI [3] PEI OPTIONAL,

 gPSI [4] GPSI OPTIONAL,

 gUTI [5] FiveGGUTI OPTIONAL,

 nRPPaMessage [6] OCTET STRING OPTIONAL,

 lPPMessage [7] OCTET STRING OPTIONAL,

 lcsCorrelationId [8] UTF8String (SIZE(1..255))

}

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

-- 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,

 uEEPSPDNConnection [19] UEEPSPDNConnection OPTIONAL,

 ePS5GSComboInfo [20] EPS5GSComboInfo OPTIONAL,

 ePSPDNConnectionEstablishment [21] EPSPDNConnectionEstablishment 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,

 pDUSessionID [11] PDUSessionID OPTIONAL,

 ePS5GSComboInfo [12] EPS5GSComboInfo 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,

 cause [10] SMFErrorCodes OPTIONAL,

 ePS5GSComboInfo [11] EPS5GSComboInfo 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,

 timeOfSessionEstablishment [19] Timestamp OPTIONAL,

 ePS5GSComboInfo [20] EPS5GSComboInfo 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

}

-- See clause 6.2.3.2.8 for details of this structure

SMFPDUtoMAPDUSessionModification ::= 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,

 pDUSessionID [11] PDUSessionID,

 requestIndication [12] RequestIndication,

 aTSSSContainer [13] ATSSSContainer

}

-- See clause 6.2.3.2.7.1 for details of this structure

SMFMAPDUSessionEstablishment ::= SEQUENCE

{

 sUPI [1] SUPI OPTIONAL,

 sUPIUnauthenticated [2] SUPIUnauthenticatedIndication OPTIONAL,

 pEI [3] PEI OPTIONAL,

 gPSI [4] GPSI OPTIONAL,

 pDUSessionID [5] PDUSessionID,

 pDUSessionType [6] PDUSessionType,

 accessInfo [7] SEQUENCE OF AccessInfo,

 sNSSAI [8] SNSSAI OPTIONAL,

 uEEndpoint [9] SEQUENCE OF UEEndpointAddress OPTIONAL,

 location [10] Location OPTIONAL,

 dNN [11] DNN,

 aMFID [12] AMFID OPTIONAL,

 hSMFURI [13] HSMFURI OPTIONAL,

 requestType [14] FiveGSMRequestType,

 sMPDUDNRequest [15] SMPDUDNRequest OPTIONAL,

 servingNetwork [16] SMFServingNetwork,

 oldPDUSessionID [17] PDUSessionID OPTIONAL,

 mAUpgradeIndication [18] SMFMAUpgradeIndication OPTIONAL,

 ePSPDNCnxInfo [19] SMFEPSPDNCnxInfo OPTIONAL,

 mAAcceptedIndication [20] SMFMAAcceptedIndication,

 aTSSSContainer [21] ATSSSContainer OPTIONAL

}

-- See clause 6.2.3.2.7.2 for details of this structure

SMFMAPDUSessionModification ::= SEQUENCE

{

 sUPI [1] SUPI OPTIONAL,

 sUPIUnauthenticated [2] SUPIUnauthenticatedIndication OPTIONAL,

 pEI [3] PEI OPTIONAL,

 gPSI [4] GPSI OPTIONAL,

 pDUSessionID [5] PDUSessionID,

 accessInfo [6] SEQUENCE OF AccessInfo OPTIONAL,

 sNSSAI [7] SNSSAI OPTIONAL,

 location [8] Location OPTIONAL,

 requestType [9] FiveGSMRequestType OPTIONAL,

 servingNetwork [10] SMFServingNetwork,

 oldPDUSessionID [11] PDUSessionID OPTIONAL,

 mAUpgradeIndication [12] SMFMAUpgradeIndication OPTIONAL,

 ePSPDNCnxInfo [13] SMFEPSPDNCnxInfo OPTIONAL,

 mAAcceptedIndication [14] SMFMAAcceptedIndication,

 aTSSSContainer [15] ATSSSContainer OPTIONAL

}

-- See clause 6.2.3.2.7.3 for details of this structure

SMFMAPDUSessionRelease ::= 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,

 cause [10] SMFErrorCodes OPTIONAL

}

-- See clause 6.2.3.2.7.4 for details of this structure

SMFStartOfInterceptionWithEstablishedMAPDUSession ::= SEQUENCE

{

 sUPI [1] SUPI OPTIONAL,

 sUPIUnauthenticated [2] SUPIUnauthenticatedIndication OPTIONAL,

 pEI [3] PEI OPTIONAL,

 gPSI [4] GPSI OPTIONAL,

 pDUSessionID [5] PDUSessionID,

 pDUSessionType [6] PDUSessionType,

 accessInfo [7] SEQUENCE OF AccessInfo,

 sNSSAI [8] SNSSAI OPTIONAL,

 uEEndpoint [9] SEQUENCE OF UEEndpointAddress OPTIONAL,

 location [10] Location OPTIONAL,

 dNN [11] DNN,

 aMFID [12] AMFID OPTIONAL,

 hSMFURI [13] HSMFURI OPTIONAL,

 requestType [14] FiveGSMRequestType OPTIONAL,

 sMPDUDNRequest [15] SMPDUDNRequest OPTIONAL,

 servingNetwork [16] SMFServingNetwork,

 oldPDUSessionID [17] PDUSessionID OPTIONAL,

 mAUpgradeIndication [18] SMFMAUpgradeIndication OPTIONAL,

 ePSPDNCnxInfo [19] SMFEPSPDNCnxInfo OPTIONAL,

 mAAcceptedIndication [20] SMFMAAcceptedIndication,

 aTSSSContainer [21] ATSSSContainer OPTIONAL

}

-- See clause 6.2.3.2.7.5 for details of this structure

SMFMAUnsuccessfulProcedure ::= SEQUENCE

{

 failedProcedureType [1] SMFFailedProcedureType,

 failureCause [2] FiveGSMCause,

 requestedSlice [3] NSSAI OPTIONAL,

 initiator [4] Initiator,

 sUPI [5] SUPI OPTIONAL,

 sUPIUnauthenticated [6] SUPIUnauthenticatedIndication OPTIONAL,

 pEI [7] PEI OPTIONAL,

 gPSI [8] GPSI OPTIONAL,

 pDUSessionID [9] PDUSessionID OPTIONAL,

 accessInfo [10] SEQUENCE OF AccessInfo,

 uEEndpoint [11] SEQUENCE OF UEEndpointAddress OPTIONAL,

 location [12] Location OPTIONAL,

 dNN [13] DNN OPTIONAL,

 aMFID [14] AMFID OPTIONAL,

 hSMFURI [15] HSMFURI OPTIONAL,

 requestType [16] FiveGSMRequestType OPTIONAL,

 sMPDUDNRequest [17] SMPDUDNRequest OPTIONAL

}

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

-- 5G SMF parameters

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

SMFID ::= UTF8String

SMFFailedProcedureType ::= ENUMERATED

{

 pDUSessionEstablishment(1),

 pDUSessionModification(2),

 pDUSessionRelease(3)

}

SMFServingNetwork ::= SEQUENCE

{

 pLMNID [1] PLMNID,

 nID [2] NID OPTIONAL

}

AccessInfo ::= SEQUENCE

{

 accessType [1] AccessType,

 rATType [2] RATType OPTIONAL,

 gTPTunnelID [3] FTEID,

 non3GPPAccessEndpoint [4] UEEndpointAddress OPTIONAL,

 establishmentStatus [5] EstablishmentStatus,

 aNTypeToReactivate [6] AccessType OPTIONAL

}

-- see Clause 6.1.2 of TS 24.193[44] for the details of the ATSSS container contents.

ATSSSContainer ::= OCTET STRING

EstablishmentStatus ::= ENUMERATED

{

 established(0),

 released(1)

}

SMFMAUpgradeIndication ::= BOOLEAN

-- Given in YAML encoding as defined in clause 6.1.6.2.31 of TS 29.502[16]

SMFEPSPDNCnxInfo ::= UTF8String

SMFMAAcceptedIndication ::= BOOLEAN

-- see Clause 6.1.6.3.8 of TS 29.502[16] for the details of this structure.

SMFErrorCodes ::= UTF8String

-- see Clause 6.1.6.3.2 of TS 29.502[16] for details of this structure.

UEEPSPDNConnection ::= OCTET STRING

-- see Clause 6.1.6.3.6 of TS 29.502[16] for the details of this structure.

RequestIndication ::= ENUMERATED

{

 uEREQPDUSESMOD(0),

 uEREQPDUSESREL(1),

 pDUSESMOB(2),

 nWREQPDUSESAUTH(3),

 nWREQPDUSESMOD(4),

 nWREQPDUSESREL(5),

 eBIASSIGNMENTREQ(6),

 rELDUETO5GANREQUEST(7)

}

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

-- PGW-C + SMF Parameters

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

CSRMFI ::= BOOLEAN

EPS5GSComboInfo ::= SEQUENCE

{

 ePSInterworkingIndication [1] EPSInterworkingIndication,

 ePSSubscriberIDs [2] EPSSubscriberIDs,

 ePSPDNCnxInfo [3] EPSPDNCnxInfo OPTIONAL,

 ePSBearerInfo [4] EPSBearerInfo OPTIONAL

}

EPSInterworkingIndication ::= ENUMERATED

{

 none(1),

 withN26(2),

 withoutN26(3),

 iwkNon3GPP(4)

}

EPSSubscriberIDs ::= SEQUENCE

{

 iMSI [1] IMSI OPTIONAL,

 mSISDN [2] MSISDN OPTIONAL,

 iMEI [3] IMEI OPTIONAL

}

EPSPDNCnxInfo ::= SEQUENCE

{

 pGWS8ControlPlaneFTEID [1] FTEID,

 linkedBearerID [2] EPSBearerID OPTIONAL

}

EPSPDNConnectionEstablishment ::= SEQUENCE

{

 ePSSubscriberIDs [1] EPSSubscriberIDs,

 iMSIUnauthenticated [2] IMSIUnauthenticatedIndication OPTIONAL,

 defaultBearerID [3] EPSBearerID,

 gTPTunnelInfo [4] GTPTunnelInfo OPTIONAL,

 pDNConnectionType [5] PDNConnectionType,

 uEEndpoints [6] SEQUENCE OF UEEndpointAddress OPTIONAL,

 non3GPPAccessEndpoint [7] UEEndpointAddress OPTIONAL,

 location [8] Location OPTIONAL,

 additionalLocation [9] Location OPTIONAL,

 aPN [10] APN,

 requestType [11] EPSPDNConnectionRequestType OPTIONAL,

 accessType [12] AccessType OPTIONAL,

 rATType [13] RATType OPTIONAL,

 protocolConfigurationOptions [14] PDNProtocolConfigurationOptions OPTIONAL,

 servingNetwork [15] SMFServingNetwork OPTIONAL,

 sMPDUDNRequest [16] SMPDUDNRequest OPTIONAL,

 bearerContextsCreated [17] SEQUENCE OF EPSBearerContextCreated,

 bearerContextsMarkedForRemoval [18] SEQUENCE OF EPSBearerContextForRemoval OPTIONAL,

 indicationFlags [19] PDNConnectionIndicationFlags OPTIONAL,

 handoverIndication [20] PDNHandoverIndication OPTIONAL,

 nBIFOMSupport [21] PDNNBIFOMSupport OPTIONAL,

 fiveGSInterworkingInfo [22] FiveGSInterworkingInfo OPTIONAL,

 cSRMFI [23] CSRMFI OPTIONAL,

 restorationOfPDNConnectionsSupport [24] RestorationOfPDNConnectionsSupport OPTIONAL,

 pGWChangeIndication [25] PGWChangeIndication OPTIONAL,

 pGWRNSI [26] PGWRNSI OPTIONAL

}

EPSBearerContextCreated ::= SEQUENCE

{

 ePSBearerID [1] EPSBearerID,

 cause [2] EPSBearerCreationCauseValue,

 gTPTunnelInfo [3] GTPTunnelInfo OPTIONAL,

 bearerQOS [4] EPSBearerQOS OPTIONAL

}

EPSBearerContextForRemoval ::= SEQUENCE

{

 ePSBearerID [1] EPSBearerID,

 cause [2] EPSBearerRemovalCauseValue

}

EPSBearerCreationCauseValue ::= INTEGER (0..255)

EPSBearerRemovalCauseValue ::= INTEGER (0..255)

EPSBearerInfo ::= SEQUENCE OF EPSBearers

EPSBearers ::= SEQUENCE

{

 ePSBearerID [1] EPSBearerID,

 pGWS8UserPlaneFTEID [2] FTEID,

 qCI [3] QCI

}

EPSBearerQOS ::= SEQUENCE

{

 qCI [1] QCI OPTIONAL,

 maximumUplinkBitRate [2] BitrateBinKBPS OPTIONAL,

 maximumDownlinkBitRate [3] BitrateBinKBPS OPTIONAL,

 guaranteedUplinkBitRate [4] BitrateBinKBPS OPTIONAL,

 guaranteedDownlinkBitRate [5] BitrateBinKBPS OPTIONAL,

 priorityLevel [6] EPSQOSPriority OPTIONAL

}

EPSQOSPriority ::= INTEGER (1..15)

BitrateBinKBPS ::= OCTET STRING

GTPTunnelInfo ::= SEQUENCE

{

 ePSGTPTunnels [1] EPSGTPTunnels

}

EPSGTPTunnels ::= SEQUENCE

{

 controlPlaneSenderFTEID [1] FTEID OPTIONAL,

 controlPlanePGWS5S8FTEID [2] FTEID OPTIONAL,

 s1UeNodeBFTEID [3] FTEID OPTIONAL,

 s5S8SGWFTEID [4] FTEID OPTIONAL,

 s5S8PGWFTEID [5] FTEID OPTIONAL,

 s2bUePDGFTEID [6] FTEID OPTIONAL,

 s2aUePDGFTEID [7] FTEID OPTIONAL

}

EPSPDNConnectionRequestType ::= ENUMERATED

{

 initialRequest(1),

 handover(2),

 rLOS(3),

 emergency(4),

 handoverOfEmergencyBearerServices(5),

 reserved(6)

}

FiveGSInterworkingInfo ::= SEQUENCE

{

 fiveGSInterworkingIndicator [1] FiveGSInterworkingIndicator,

 fiveGSInterworkingWithoutN26 [2] FiveGSInterworkingWithoutN26,

 fiveGCNotRestrictedSupport [3] FiveGCNotRestrictedSupport

}

FiveGSInterworkingIndicator ::= BOOLEAN

FiveGSInterworkingWithoutN26 ::= BOOLEAN

FiveGCNotRestrictedSupport ::= BOOLEAN

PDNConnectionIndicationFlags ::= OCTET STRING

PDNHandoverIndication ::= BOOLEAN

PDNNBIFOMSupport ::= BOOLEAN

PDNProtocolConfigurationOptions ::= SEQUENCE

{

 requestPCO [1] PDNPCO OPTIONAL,

 requestAPCO [2] PDNPCO OPTIONAL,

 requestEPCO [3] PDNPCO OPTIONAL,

 responsePCO [4] PDNPCO OPTIONAL,

 responseAPCO [5] PDNPCO OPTIONAL,

 responseEPCO [6] PDNPCO OPTIONAL

}

PDNPCO ::= OCTET STRING

PGWChangeIndication ::= BOOLEAN

PGWRNSI ::= BOOLEAN

QCI ::= INTEGER (0..255)

RestorationOfPDNConnectionsSupport ::= BOOLEAN

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

-- 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

}

UDMSubscriberRecordChangeMessage ::= SEQUENCE

{

 sUPI [1] SUPI OPTIONAL,

 pEI [2] PEI OPTIONAL,

 gPSI [3] GPSI OPTIONAL,

 oldPEI [4] PEI OPTIONAL,

 oldSUPI [5] SUPI OPTIONAL,

 oldGPSI [6] GPSI OPTIONAL,

 oldserviceID [7] ServiceID OPTIONAL,

 subscriberRecordChangeMethod [8] UDMSubscriberRecordChangeMethod,

 serviceID [9] ServiceID OPTIONAL

}

UDMCancelLocationMessage ::= SEQUENCE

{

 sUPI [1] SUPI,

 pEI [2] PEI OPTIONAL,

 gPSI [3] GPSI OPTIONAL,

 gUAMI [4] GUAMI OPTIONAL,

 pLMNID [5] PLMNID OPTIONAL,

 cancelLocationMethod [6] UDMCancelLocationMethod

}

UDMLocationInformationResult ::= SEQUENCE

{

 sUPI [1] SUPI,

 pEI [2] PEI OPTIONAL,

 gPSI [3] GPSI OPTIONAL,

 locationInfoRequest [4] UDMLocationInfoRequest,

 vPLMNID [5] PLMNID OPTIONAL,

 currentLocationIndicator [6] BOOLEAN OPTIONAL,

 aMFInstanceID [7] NFID OPTIONAL,

 sMSFInstanceID [8] NFID OPTIONAL,

 location [9] Location OPTIONAL,

 rATType [10] RATType OPTIONAL,

 problemDetails [11] UDMProblemDetails OPTIONAL

}

UDMUEInformationResponse ::= SEQUENCE

{

 sUPI [1] SUPI,

 tADSInfo [2] UEContextInfo OPTIONAL,

 fiveGSUserStateInfo [3] FiveGSUserStateInfo OPTIONAL,

 fiveGSRVCCInfo [4] FiveGSRVCCInfo OPTIONAL,

 problemDetails [5] UDMProblemDetails OPTIONAL

}

UDMUEAuthenticationResponse ::= SEQUENCE

{

 sUPI [1] SUPI,

 authenticationInfoRequest [2] UDMAuthenticationInfoRequest,

 aKMAIndicator [3] BOOLEAN OPTIONAL,

 problemDetails [4] UDMProblemDetails OPTIONAL

}

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

-- 5G UDM parameters

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

UDMServingSystemMethod ::= ENUMERATED

{

 amf3GPPAccessRegistration(0),

 amfNon3GPPAccessRegistration(1),

 unknown(2)

}

UDMSubscriberRecordChangeMethod ::= ENUMERATED

{

 pEIChange(1),

 sUPIChange(2),

 gPSIChange(3),

 uEDeprovisioning(4),

 unknown(5),

 serviceIDChange(6)

}

UDMCancelLocationMethod ::= ENUMERATED

{

 aMF3GPPAccessDeregistration(1),

 aMFNon3GPPAccessDeregistration(2),

 uDMDeregistration(3),

 unknown(4)

}

ServiceID ::= SEQUENCE

{

 nSSAI [1] NSSAI OPTIONAL,

 cAGID [2] SEQUENCE OF CAGID OPTIONAL

}

CAGID ::= UTF8String

UDMAuthenticationInfoRequest ::= SEQUENCE

{

 infoRequestType [1] UDMInfoRequestType,

 rGAuthCtx [2] SEQUENCE SIZE(1..MAX) OF SubscriberIdentifier,

 authType [3] PrimaryAuthenticationType,

 servingNetworkName [4] PLMNID,

 aUSFInstanceID [5] NFID OPTIONAL,

 cellCAGInfo [6] CAGID OPTIONAL,

 n5GCIndicator [7] BOOLEAN OPTIONAL

}

UDMLocationInfoRequest ::= SEQUENCE

{

 requested5GSLocation [1] BOOLEAN OPTIONAL,

 requestedCurrentLocation [2] BOOLEAN OPTIONAL,

 requestedRATType [3] BOOLEAN OPTIONAL,

 requestedTimeZone [4] BOOLEAN OPTIONAL,

 requestedServingNode [5] BOOLEAN OPTIONAL

}

UDMProblemDetails ::= SEQUENCE

{

 cause [1] UDMProblemDetailsCause OPTIONAL

}

UDMProblemDetailsCause ::= CHOICE

{

 uDMDefinedCause [1] UDMDefinedCause,

 otherCause [2] UDMProblemDetailsOtherCause

}

UDMDefinedCause ::= ENUMERATED

{

 userNotFound(1),

 dataNotFound(2),

 contextNotFound(3),

 subscriptionNotFound(4),

 other(5)

}

UDMInfoRequestType ::= ENUMERATED

{

 hSS(1),

 aUSF(2),

 other(3)

}

UDMProblemDetailsOtherCause ::= SEQUENCE

{

 problemDetailsType [1] UTF8String OPTIONAL,

 title [2] UTF8String OPTIONAL,

 status [3] INTEGER OPTIONAL,

 detail [4] UTF8String OPTIONAL,

 instance [5] UTF8String OPTIONAL,

 cause [6] UTF8String OPTIONAL,

 uDMInvalidParameters [7] UDMInvalidParameters,

 uDMSupportedFeatures [8] UTF8String

}

UDMInvalidParameters ::= SEQUENCE

{

 parameter [1] UTF8String OPTIONAL,

 reason [2] UTF8String OPTIONAL

}

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

-- 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,

 linkTransferStatus [4] SMSTransferStatus,

 otherMessage [5] SMSOtherMessageIndication OPTIONAL,

 location [6] Location OPTIONAL,

 peerNFAddress [7] SMSNFAddress OPTIONAL,

 peerNFType [8] SMSNFType OPTIONAL,

 sMSTPDUData [9] SMSTPDUData OPTIONAL,

 messageType [10] SMSMessageType OPTIONAL,

 rPMessageReference [11] SMSRPMessageReference OPTIONAL

}

SMSReport ::= SEQUENCE

{

 location [1] Location OPTIONAL,

 sMSTPDUData [2] SMSTPDUData,

 messageType [3] SMSMessageType,

 rPMessageReference [4] SMSRPMessageReference

}

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

-- 5G SMSF parameters

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

SMSAddress ::= OCTET STRING(SIZE(2..12))

SMSMessageType ::= ENUMERATED

{

 deliver(1),

 deliverReportAck(2),

 deliverReportError(3),

 statusReport(4),

 command(5),

 submit(6),

 submitReportAck(7),

 submitReportError(8),

 reserved(9)

}

SMSParty ::= SEQUENCE

{

 sUPI [1] SUPI OPTIONAL,

 pEI [2] PEI OPTIONAL,

 gPSI [3] GPSI OPTIONAL,

 sMSAddress [4] SMSAddress 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)

}

SMSRPMessageReference ::= INTEGER (0..255)

SMSTPDUData ::= CHOICE

{

 sMSTPDU [1] SMSTPDU,

 truncatedSMSTPDU [2] TruncatedSMSTPDU

}

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

TruncatedSMSTPDU ::= OCTET STRING (SIZE(1..130))

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

-- 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

}

-- =========

-- PTC CCPDU

-- =========

PTCCCPDU ::= OCTET STRING

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

-- 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)

}

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

-- IMS definitions

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

-- See clause 7.12.4.2.1 for details of this structure

IMSMessage ::= SEQUENCE

{

 payload [1] IMSPayload,

 sessionDirection [2] SessionDirection,

 voIPRoamingIndication [3] VoIPRoamingIndication OPTIONAL,

 location [6] Location OPTIONAL

}

-- See clause 7.12.4.2.2 for details of this structure

StartOfInterceptionForActiveIMSSession ::= SEQUENCE

{

 originatingId [1] SEQUENCE OF IMPU,

 terminatingId [2] IMPU,

 sDPState [3] SEQUENCE OF OCTET STRING OPTIONAL,

 diversionIdentity [4] IMPU OPTIONAL,

 voIPRoamingIndication [5] VoIPRoamingIndication OPTIONAL,

 location [7] Location OPTIONAL

}

-- See clause 7.12.4.2.3 for the details.

IMSCCUnavailable ::= SEQUENCE

{

 cCUnavailableReason [1] UTF8String,

 sDPState [2] OCTET STRING OPTIONAL

}

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

-- IMS parameters

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

IMSPayload ::= CHOICE

{

 encapsulatedSIPMessage [1] SIPMessage

}

SIPMessage ::= SEQUENCE

{

 iPSourceAddress [1] IPAddress,

 iPDestinationAddress [2] IPAddress,

 sIPContent [3] OCTET STRING

}

VoIPRoamingIndication ::= ENUMERATED

{

 roamingLBO(1),

 roamingS8HR(2),

 roamingN9HR(3)

}

SessionDirection ::= ENUMERATED

{

 fromTarget(1),

 toTarget(2),

 combined(3),

 indeterminate(4)

}

HeaderOnlyIndication ::= BOOLEAN

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

-- STIR/SHAKEN/RCD/eCNAM definitions

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

-- See clause 7.11.2.1.2 for details of this structure

STIRSHAKENSignatureGeneration ::= SEQUENCE

{

 pASSporTs [1] SEQUENCE OF PASSporT,

 encapsulatedSIPMessage [2] SIPMessage OPTIONAL

}

-- See clause 7.11.2.1.3 for details of this structure

STIRSHAKENSignatureValidation ::= SEQUENCE

{

 pASSporTs [1] SEQUENCE OF PASSporT OPTIONAL,

 rCDTerminalDisplayInfo [2] RCDDisplayInfo OPTIONAL,

 eCNAMTerminalDisplayInfo [3] ECNAMDisplayInfo OPTIONAL,

 sHAKENValidationResult [4] SHAKENValidationResult,

 sHAKENFailureStatusCode [5] SHAKENFailureStatusCode OPTIONAL,

 encapsulatedSIPMessage [6] SIPMessage OPTIONAL

}

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

-- STIR/SHAKEN/RCD/eCNAM parameters

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

PASSporT ::= SEQUENCE

{

 pASSporTHeader [1] PASSporTHeader,

 pASSporTPayload [2] PASSporTPayload,

 pASSporTSignature [3] OCTET STRING

}

PASSporTHeader ::= SEQUENCE

{

 type [1] JWSTokenType,

 algorithm [2] UTF8String,

 ppt [3] UTF8String OPTIONAL,

 x5u [4] UTF8String

}

JWSTokenType ::= ENUMERATED

{

 passport(1)

}

PASSporTPayload ::= SEQUENCE

{

 issuedAtTime [1] GeneralizedTime,

 originator [2] STIRSHAKENOriginator,

 destination [3] STIRSHAKENDestinations,

 attestation [4] Attestation,

 origId [5] UTF8String,

 diversion [6] STIRSHAKENDestination

}

STIRSHAKENOriginator ::= CHOICE

{

 telephoneNumber [1] STIRSHAKENTN,

 sTIRSHAKENURI [2] UTF8String

}

STIRSHAKENDestinations ::= SEQUENCE OF STIRSHAKENDestination

STIRSHAKENDestination ::= CHOICE

{

 telephoneNumber [1] STIRSHAKENTN,

 sTIRSHAKENURI [2] UTF8String

}

STIRSHAKENTN ::= CHOICE

{

 mSISDN [1] MSISDN

}

Attestation ::= ENUMERATED

{

 attestationA(1),

 attestationB(2),

 attestationC(3)

}

SHAKENValidationResult ::= ENUMERATED

{

 tNValidationPassed(1),

 tNValidationFailed(2),

 noTNValidation(3)

}

SHAKENFailureStatusCode ::= INTEGER

ECNAMDisplayInfo ::= SEQUENCE

{

 name [1] UTF8String,

 additionalInfo [2] OCTET STRING OPTIONAL

}

RCDDisplayInfo ::= SEQUENCE

{

 name [1] UTF8String,

 jcd [2] OCTET STRING OPTIONAL,

 jcl [3] OCTET STRING OPTIONAL

}

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

-- 5G LALS definitions

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

LALSReport ::= SEQUENCE

{

 sUPI [1] SUPI OPTIONAL,

-- pEI [2] PEI OPTIONAL, deprecated in Release-16, do not re-use this tag number

 gPSI [3] GPSI OPTIONAL,

 location [4] Location OPTIONAL,

 iMPU [5] IMPU OPTIONAL,

 iMSI [7] IMSI OPTIONAL,

 mSISDN [8] MSISDN 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),

 startOfFlow(4),

 endOfFlow(5)

}

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

-- Identifier Association definitions

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

AMFIdentifierAssociation ::= SEQUENCE

{

 sUPI [1] SUPI,

 sUCI [2] SUCI OPTIONAL,

 pEI [3] PEI OPTIONAL,

 gPSI [4] GPSI OPTIONAL,

 gUTI [5] FiveGGUTI,

 location [6] Location,

 fiveGSTAIList [7] TAIList OPTIONAL

}

MMEIdentifierAssociation ::= SEQUENCE

{

 iMSI [1] IMSI,

 iMEI [2] IMEI OPTIONAL,

 mSISDN [3] MSISDN OPTIONAL,

 gUTI [4] GUTI,

 location [5] Location,

 tAIList [6] TAIList OPTIONAL

}

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

-- Identifier Association parameters

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

MMEGroupID ::= OCTET STRING (SIZE(2))

MMECode ::= OCTET STRING (SIZE(1))

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

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

-- EPS MME definitions

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

MMEAttach ::= SEQUENCE

{

 attachType [1] EPSAttachType,

 attachResult [2] EPSAttachResult,

 iMSI [3] IMSI,

 iMEI [4] IMEI OPTIONAL,

 mSISDN [5] MSISDN OPTIONAL,

 gUTI [6] GUTI OPTIONAL,

 location [7] Location OPTIONAL,

 ePSTAIList [8] TAIList OPTIONAL,

 sMSServiceStatus [9] EPSSMSServiceStatus OPTIONAL,

 oldGUTI [10] GUTI OPTIONAL,

 eMM5GRegStatus [11] EMM5GMMStatus OPTIONAL

}

MMEDetach ::= SEQUENCE

{

 detachDirection [1] MMEDirection,

 detachType [2] EPSDetachType,

 iMSI [3] IMSI,

 iMEI [4] IMEI OPTIONAL,

 mSISDN [5] MSISDN OPTIONAL,

 gUTI [6] GUTI OPTIONAL,

 cause [7] EMMCause OPTIONAL,

 location [8] Location OPTIONAL,

 switchOffIndicator [9] SwitchOffIndicator OPTIONAL

}

MMELocationUpdate ::= SEQUENCE

{

 iMSI [1] IMSI,

 iMEI [2] IMEI OPTIONAL,

 mSISDN [3] MSISDN OPTIONAL,

 gUTI [4] GUTI OPTIONAL,

 location [5] Location OPTIONAL,

 oldGUTI [6] GUTI OPTIONAL,

 sMSServiceStatus [7] EPSSMSServiceStatus OPTIONAL

}

MMEStartOfInterceptionWithEPSAttachedUE ::= SEQUENCE

{

 attachType [1] EPSAttachType,

 attachResult [2] EPSAttachResult,

 iMSI [3] IMSI,

 iMEI [4] IMEI OPTIONAL,

 mSISDN [5] MSISDN OPTIONAL,

 gUTI [6] GUTI OPTIONAL,

 location [7] Location OPTIONAL,

 ePSTAIList [9] TAIList OPTIONAL,

 sMSServiceStatus [10] EPSSMSServiceStatus OPTIONAL,

 eMM5GRegStatus [12] EMM5GMMStatus OPTIONAL

}

MMEUnsuccessfulProcedure ::= SEQUENCE

{

 failedProcedureType [1] MMEFailedProcedureType,

 failureCause [2] MMEFailureCause,

 iMSI [3] IMSI OPTIONAL,

 iMEI [4] IMEI OPTIONAL,

 mSISDN [5] MSISDN OPTIONAL,

 gUTI [6] GUTI OPTIONAL,

 location [7] Location OPTIONAL

}

-- See clause 6.3.2.2.8 for details of this structure

MMEPositioningInfoTransfer ::= SEQUENCE

{

 iMSI [1] IMSI,

 iMEI [2] IMEI OPTIONAL,

 mSISDN [3] MSISDN OPTIONAL,

 gUTI [4] GUTI OPTIONAL,

 lPPaMessage [5] OCTET STRING OPTIONAL,

 lPPMessage [6] OCTET STRING OPTIONAL,

 mMELCSCorrelationId [7] OCTET STRING (SIZE(4))

}

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

-- EPS MME parameters

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

EMMCause ::= INTEGER (0..255)

ESMCause ::= INTEGER (0..255)

EPSAttachType ::= ENUMERATED

{

 ePSAttach(1),

 combinedEPSIMSIAttach(2),

 ePSRLOSAttach(3),

 ePSEmergencyAttach(4),

 reserved(5)

}

EPSAttachResult ::= ENUMERATED

{

 ePSOnly(1),

 combinedEPSIMSI(2)

}

EPSDetachType ::= ENUMERATED

{

 ePSDetach(1),

 iMSIDetach(2),

 combinedEPSIMSIDetach(3),

 reAttachRequired(4),

 reAttachNotRequired(5),

 reserved(6)

}

EPSSMSServiceStatus ::= ENUMERATED

{

 sMSServicesNotAvailable(1),

 sMSServicesNotAvailableInThisPLMN(2),

 networkFailure(3),

 congestion(4)

}

MMEDirection ::= ENUMERATED

{

 networkInitiated(1),

 uEInitiated(2)

}

MMEFailedProcedureType ::= ENUMERATED

{

 attachReject(1),

 authenticationReject(2),

 securityModeReject(3),

 serviceReject(4),

 trackingAreaUpdateReject(5),

 activateDedicatedEPSBearerContextReject(6),

 activateDefaultEPSBearerContextReject(7),

 bearerResourceAllocationReject(8),

 bearerResourceModificationReject(9),

 modifyEPSBearerContextReject(10),

 pDNConnectivityReject(11),

 pDNDisconnectReject(12)

}

MMEFailureCause ::= CHOICE

{

 eMMCause [1] EMMCause,

 eSMCause [2] ESMCause

}

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

-- 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

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

-- 5G EPS Interworking Parameters

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

EMM5GMMStatus ::= SEQUENCE

{

 eMMRegStatus [1] EMMRegStatus OPTIONAL,

 fiveGMMStatus [2] FiveGMMStatus OPTIONAL

}

EPS5GGUTI ::= CHOICE

{

 gUTI [1] GUTI,

 fiveGGUTI [2] FiveGGUTI

}

EMMRegStatus ::= ENUMERATED

{

 uEEMMRegistered(1),

 uENotEMMRegistered(2)

}

FiveGMMStatus ::= ENUMERATED

{

 uE5GMMRegistered(1),

 uENot5GMMRegistered(2)

}

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

-- Separated Location Reporting definitions

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

SeparatedLocationReporting ::= SEQUENCE

{

 sUPI [1] SUPI,

 sUCI [2] SUCI OPTIONAL,

 pEI [3] PEI OPTIONAL,

 gPSI [4] GPSI OPTIONAL,

 gUTI [5] FiveGGUTI OPTIONAL,

 location [6] Location,

 non3GPPAccessEndpoint [7] UEEndpointAddress OPTIONAL,

 rATType [8] RATType OPTIONAL

}

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

-- 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

EUI64 ::= OCTET STRING (SIZE(8))

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)

FiveGSRVCCInfo ::= SEQUENCE

{

 uE5GSRVCCCapability [1] BOOLEAN,

 sessionTransferNumber [2] UTF8String OPTIONAL,

 correlationMSISDN [3] MSISDN OPTIONAL

}

FiveGSUserStateInfo ::= SEQUENCE

{

 fiveGSUserState [1] FiveGSUserState,

 accessType [2] AccessType

}

FiveGSUserState ::= ENUMERATED

{

 deregistered(1),

 registeredNotReachableForPaging(2),

 registeredReachableForPaging(3),

 connectedNotReachableForPaging(4),

 connectedReachableForPaging(5),

 notProvidedFromAMF(6)

}

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

}

GUTI ::= SEQUENCE

{

 mCC [1] MCC,

 mNC [2] MNC,

 mMEGroupID [3] MMEGroupID,

 mMECode [4] MMECode,

 mTMSI [5] TMSI

}

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))

IMSIUnauthenticatedIndication ::= BOOLEAN

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))

MACRestrictionIndicator ::= ENUMERATED

{

 noResrictions(1),

 mACAddressNotUseableAsEquipmentIdentifier(2),

 unknown(3)

}

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)

}

NonIMEISVPEI ::= CHOICE

{

 mACAddress [1] MACAddress

}

NSSAI ::= SEQUENCE OF SNSSAI

PLMNID ::= SEQUENCE

{

 mCC [1] MCC,

 mNC [2] MNC

}

PDNConnectionType ::= ENUMERATED

{

 iPv4(1),

 iPv6(2),

 iPv4v6(3),

 nonIP(4),

 ethernet(5)

}

PDUSessionID ::= INTEGER (0..255)

PDUSessionType ::= ENUMERATED

{

 iPv4(1),

 iPv6(2),

 iPv4v6(3),

 unstructured(4),

 ethernet(5)

}

PEI ::= CHOICE

{

 iMEI [1] IMEI,

 iMEISV [2] IMEISV,

 mACAddress [3] MACAddress,

 eUI64 [4] EUI64

}

PortNumber ::= INTEGER(0..65535)

PrimaryAuthenticationType ::= ENUMERATED

{

 eAPAKAPrime(1),

 fiveGAKA(2),

 eAPTLS(3),

 none(4),

 ePSAKA(5),

 eAPAKA(6),

 iMSAKA(7),

 gBAAKA(8),

 uMTSAKA(9)

}

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),

 nRLEO(16),

 nRMEO(17),

 nRGEO(18),

 nROTHERSAT(19),

 nRREDCAP(20)

}

RejectedNSSAI ::= SEQUENCE OF RejectedSNSSAI

RejectedSNSSAI ::= SEQUENCE

{

 causeValue [1] RejectedSliceCauseValue,

 sNSSAI [2] SNSSAI

}

RejectedSliceCauseValue ::= INTEGER (0..255)

ReRegRequiredIndicator ::= ENUMERATED

{

 reRegistrationRequired(1),

 reRegistrationNotRequired(2)

}

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

-- TS 24.501 [13], clause 9.11.3.6.1

SMSOverNASIndicator ::= ENUMERATED

{

 sMSOverNASNotAllowed(1),

 sMSOverNASAllowed(2)

}

SNSSAI ::= SEQUENCE

{

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

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

}

SubscriberIdentifier ::= CHOICE

{

 sUCI [1] SUCI,

 sUPI [2] SUPI

}

SUCI ::= SEQUENCE

{

 mCC [1] MCC,

 mNC [2] MNC,

 routingIndicator [3] RoutingIndicator,

 protectionSchemeID [4] ProtectionSchemeID,

 homeNetworkPublicKeyID [5] HomeNetworkPublicKeyID,

 schemeOutput [6] SchemeOutput,

 routingIndicatorLength [7] INTEGER (1..4) OPTIONAL

 -- shall be included if different from the number of meaningful digits given

 -- in routingIndicator

}

SUPI ::= CHOICE

{

 iMSI [1] IMSI,

 nAI [2] NAI

}

SUPIUnauthenticatedIndication ::= BOOLEAN

SwitchOffIndicator ::= ENUMERATED

{

 normalDetach(1),

 switchOff(2)

}

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

UEContextInfo ::= SEQUENCE

{

 supportVoPS [1] BOOLEAN OPTIONAL,

 supportVoPSNon3GPP [2] BOOLEAN OPTIONAL,

 lastActiveTime [3] Timestamp OPTIONAL,

 accessType [4] AccessType OPTIONAL,

 rATType [5] RATType OPTIONAL

}

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,

 ePSLocationInfo [4] EPSLocationInfo 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,

 ageOfLocationInfo [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,

 ageOfLocationInfo [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,

 tNAPID [5] TNAPID OPTIONAL,

 tWAPID [6] TWAPID OPTIONAL,

 hFCNodeID [7] HFCNodeID OPTIONAL,

 gLI [8] GLI OPTIONAL,

 w5GBANLineType [9] W5GBANLineType OPTIONAL,

 gCI [10] GCI OPTIONAL,

 ageOfLocationInfo [11] INTEGER OPTIONAL,

 uELocationTimestamp [12] Timestamp OPTIONAL,

 protocol [13] TransportProtocol 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,

 wAGFID [5] WAGFID,

 tNGFID [6] TNGFID

}

-- 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

}

CGI ::= SEQUENCE

{

 lAI [1] LAI,

 cellID [2] CellID

}

LAI ::= SEQUENCE

{

 pLMNID [1] PLMNID,

 lAC [2] LAC

}

LAC ::= OCTET STRING (SIZE(2))

CellID ::= OCTET STRING (SIZE(2))

SAI ::= SEQUENCE

{

 pLMNID [1] PLMNID,

 lAC [2] LAC,

 sAC [3] SAC

}

SAC ::= OCTET STRING (SIZE(2))

-- TS 29.571 [17], clause 5.4.4.5

ECGI ::= SEQUENCE

{

 pLMNID [1] PLMNID,

 eUTRACellID [2] EUTRACellID,

 nID [3] NID OPTIONAL

}

TAIList ::= SEQUENCE OF TAI

-- 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], clause 5.4.4.28 and table 5.4.2-1

TNGFID ::= UTF8String

-- TS 29.571 [17], clause 5.4.4.28 and table 5.4.2-1

WAGFID ::= UTF8String

-- TS 29.571 [17], clause 5.4.4.62

TNAPID ::= SEQUENCE

{

 sSID [1] SSID OPTIONAL,

 bSSID [2] BSSID OPTIONAL,

 civicAddress [3] CivicAddressBytes OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.64

TWAPID ::= SEQUENCE

{

 sSID [1] SSID OPTIONAL,

 bSSID [2] BSSID OPTIONAL,

 civicAddress [3] CivicAddressBytes OPTIONAL

}

-- TS 29.571 [17], clause 5.4.4.62 and clause 5.4.4.64

SSID ::= UTF8String

-- TS 29.571 [17], clause 5.4.4.62 and clause 5.4.4.64

BSSID ::= UTF8String

-- TS 29.571 [17], clause 5.4.4.36 and table 5.4.2-1

HFCNodeID ::= UTF8String

-- TS 29.571 [17], clause 5.4.4.10 and table 5.4.2-1

-- Contains the original binary data i.e. value of the YAML field after base64 encoding is removed

GLI ::= OCTET STRING (SIZE(0..150))

-- TS 29.571 [17], clause 5.4.4.10 and table 5.4.2-1

GCI ::= UTF8String

-- TS 29.571 [17], clause 5.4.4.10 and table 5.4.3.38

TransportProtocol ::= ENUMERATED

{

 uDP(1),

 tCP(2)

}

-- TS 29.571 [17], clause 5.4.4.10 and clause 5.4.3.33

W5GBANLineType ::= ENUMERATED

{

 dSL(1),

 pON(2)

}

-- 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 29.172 [53], table 6.2.2-2

EPSLocationInfo ::= SEQUENCE

{

 locationData [1] LocationData,

 cGI [2] CGI OPTIONAL,

 sAI [3] SAI OPTIONAL,

 eSMLCCellInfo [4] ESMLCCellInfo OPTIONAL

}

-- TS 29.172 [53], clause 7.4.57

ESMLCCellInfo ::= SEQUENCE

{

 eCGI [1] ECGI,

 cellPortionID [2] CellPortionID

}

-- TS 29.171 [54], clause 7.4.31

CellPortionID ::= INTEGER (0..4095)

-- 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 29.571 [17], clauses 5.4.4.62 and 5.4.4.64

-- Contains the original binary data i.e. value of the YAML field after base64 encoding is removed

CivicAddressBytes ::= OCTET STRING

-- TS 29.572 [24], clause 6.1.6.2.15

PositioningMethodAndUsage ::= SEQUENCE

{

 method [1] PositioningMethod,

 mode [2] PositioningMode,

 usage [3] Usage,

 methodCode [4] MethodCode OPTIONAL

}

-- 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),

 barometricPressure(4),

 wLAN(5),

 bluetooth(6),

 mBS(7),

 motionSensor(8),

 dLTDOA(9),

 dLAOD(10),

 multiRTT(11),

 nRECID(12),

 uLTDOA(13),

 uLAOA(14),

 networkSpecific(15)

}

-- 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),

 bDS(7),

 nAVIC(8)

}

-- 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

-- TS 29.572 [24], clause 6.1.6.2.15

MethodCode ::= INTEGER (16..31)

END

\*\*\* End of All Changes \*\*\*