**3GPP TSG-SA3 Meeting #88-LI-e-a *s3i230075***

**Online, , 23rd Jan 2023 - 27th Jan 2023**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.2* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **33.128** | **CR** | **0490** | **rev** | **1** | **Current version:** | **18.2.0** |  |
|  | | | | | | | | |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* | | | | | | | | |
|  | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME |  | Radio Access Network |  | Core Network | **X** |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** |  | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | SA3-LI (OTD) | | | | | | | | | |
| ***Source to TSG:*** |  | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | LI18 | | | | |  | ***Date:*** | | |  |
|  |  | | | |  | |  | | |  |
| ***Category:*** | **C** |  | | | | | ***Release:*** | | | Rel-18 |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)*  Detailed explanations of the above categories can be found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) … Rel-16 (Release 16) Rel-17 (Release 17) Rel-18 (Release 18) Rel-19 (Release 19)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | There are times when it is beneficial to reuse structures defined in other documents within LI reports. In many cases, this can be done by sending encapsulated information. However, in order to be able to remove unauthorized information a flexible encapsulation structure is needed. In addition, there needs to be a way to signal what the content type or schema of the payload in order to allow it to be properly decoded. This CR proposes a solution for the flexible reporting of MIME Entities and MSRP Messages. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | This CR proposes a dynamic structure that allows for the encapsulation of portions of MIME Messages and MSRP Messages and a clear way to remove unauthorized fields from the header or content. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | There will not be a solution to send portions of data. In this CR and a reliant CR, this structure is proposed to be used for RCS and for IMS message reporting. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 2, 7.X | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **Y** | **N** |  | | | |  | | |
| ***Other specs*** | | **X** |  | Other core specifications | | | | TS 33.128 CR 0489 | | |
| ***affected:*** | |  | **X** | Test specifications | | | | TS/TR ... CR ... | | |
| ***(show related CRs)*** | |  | **X** | O&M Specifications | | | | TS/TR ... CR ... | | |
|  | |  | | | | | | | | |
| ***Other comments:*** | | This CR is associated with the following changes in the Forge:  Merge request: [!139](https://forge.3gpp.org/rep/sa3/li/-/merge_requests/139)  Commit hash: [4e90b88837f0dbf34791efb170804640a85f3ec4](https://forge.3gpp.org/rep/sa3/li/-/merge_requests/139/diffs?commit_id=4e90b88837f0dbf34791efb170804640a85f3ec4)  The structures added by this CR are used by CR 0489. The structures are present in the ASN.1 the forge branch for that CR to allow it to compile correctly. | | | | | | | | |
|  | |  | | | | | | | | |
| ***This CR's revision history:*** | | S3i230058 | | | | | | | | |

## \*\*\*\* START OF FIRST CHANGE (MAIN DOCUMENT) \*\*\*

# 2 References

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

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

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

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

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

[2] 3GPP TS 23.501: "System Architecture for the 5G System".

[3] 3GPP TS 33.126: "Lawful Interception Requirements".

[4] 3GPP TS 23.502: "Procedures for the 5G System; Stage 2".

[5] 3GPP TS 33.127: "Lawful Interception (LI) Architecture and Functions".

[6] ETSI TS 103 120: "Lawful Interception (LI); Interface for warrant information".

[7] ETSI TS 103 221-1: "Lawful Interception (LI); Internal Network Interfaces; Part 1: X1".

[8] ETSI TS 103 221-2: "Lawful Interception (LI); Internal Network Interfaces; Part 2: X2/X3".

[9] ETSI TS 102 232-1: "Lawful Interception (LI); Handover Interface and Service-Specific Details (SSD) for IP delivery; Part 1: Handover specification for IP delivery".

[10] ETSI TS 102 232-7: "Lawful Interception (LI); Handover Interface and Service-Specific Details (SSD) for IP delivery; Part 7: Service-specific details for Mobile Services".

[11] 3GPP TS 33.501: "Security Architecture and Procedures for the 5G System".

[12] 3GPP TS 33.108: "3G security; Handover interface for Lawful Interception (LI)".

[13] 3GPP TS 24.501: "Non-Access-Stratum (NAS) protocol for 5G System (5GS)".

[14] 3GPP TS 24.007: "Mobile radio interface signalling layer 3; General Aspects".

[15] 3GPP TS 29.244: "Interface between the Control Plane and the User Plane nodes".

[16] 3GPP TS 29.502: "5G System; Session Management Services; Stage 3".

[17] 3GPP TS 29.571: "5G System; Common Data Types for Service Based Interfaces; Stage 3".

[18] 3GPP TS 23.040: "Technical realization of the Short Message Service (SMS)".

[19] 3GPP TS 23.003: "Numbering, addressing and identification ".

[20] OMA-TS-MLP-V3\_5-20181211-C: "Open Mobile Alliance; Mobile Location Protocol, Candidate Version 3.5", <https://www.openmobilealliance.org/release/MLS/V1_4-20181211-C/OMA-TS-MLP-V3_5-20181211-C.pdf>.

[21] 3GPP TS 29.540: "5G System; SMS Services; Stage 3".

[22] 3GPP TS 29.518: "5G System; Access and Mobility Management Services; Stage 3".

[23] 3GPP TS 38.413: "NG Application Protocol (NGAP)".

[24] 3GPP TS 29.572: "Location Management Services; Stage 3".

[25] 3GPP TS 29.503: "5G System; Unified Data Management Services".

[26] IETF RFC 815: "IP datagram reassembly algorithms".

[27] IETF RFC 2460: "Internet Protocol, Version 6 (IPv6) Specification".

[28] IETF RFC 793: "Transmission Control Protocol".

[29] IETF RFC 768: "User Datagram Protocol".

[30] IETF RFC 4340: "Datagram Congestion Control Protocol (DCCP)".

[31] IETF RFC 4960: "Stream Control Transmission Protocol".

[32] IANA (www.iana.org): Assigned Internet Protocol Numbers, "Protocol Numbers".

[33] IETF RFC 6437: "IPv6 Flow Label Specification".

[34] IETF RFC 791: "Internet Protocol".

[35] Open Geospatial Consortium OGC 05-010: "URNs of definitions in ogc namespace".

[36] 3GPP TS 33.107: "3G security; Lawful interception architecture and functions".

[37] 3GPP TS 37.340: "Evolved Universal Radio Access (E-UTRA) and NR-Multi-connectivity; Stage 2".

[38] 3GPP TS 36.413: "S1 Application Protocol (S1AP)".

[39] OMA-TS-MMS\_ENC-V1\_3-20110913-A: "Multimedia Messaging Service Encapsulation Protocol".

[40] 3GPP TS 23.140: "Multimedia Messaging Protocol. Functional Description. Stage 2".

[41] 3GPP TS 38.415: "NG-RAN; PDU Session User Plane Protocol".

[42] 3GPP TS 23.273: "5G System (5GS) Location Services (LCS); Stage 2".

[43] IETF RFC 4566: "SDP: Session Description Protocol".

[44] 3GPP TS 24.193: "Stage 3: Access Traffic Steering, Switching and Splitting (ATSSS)".

[45] 3GPP TS 29.509: "5G System; Authentication Server Services; Stage 3".

[46] 3GPP TS 24.011: "Point-to-Point (PP) Short Message Service (SMS) support on mobile radio interface".

[47] 3GPP TS 29.002: "Mobile Application Part (MAP) specification".

[48] 3GPP TS 29.504: "5G System; Unified Data Repository Services; Stage 3".

[49] 3GPP TS 29.505: "5G System; Usage of the Unified Data Repository services for Subscription Data; Stage 3".

[50] 3GPP TS 23.401 "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access".

[51] 3GPP TS 24.301 "Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS), Stage 3".

[52] 3GPP TS 23.271 "Functional stage 2 description of Location Services (LCS)".

[53] 3GPP TS 29.172 "Evolved Packet Core (EPC) LCS Protocol (ELP) between the Gateway Mobile Location Centre (GMLC) and the Mobile Management Entity (MME); SLg interface".

[54] 3GPP TS 29.171 "LCS Application Protocol (LCS-AP) between the Mobile Management Entity (MME) and Evolved Serving Mobile Location Centre (E-SMLC); SLs interface".

[55] 3GPP TS 24.379: "Mission Critical Push to Talk (MCPTT) call control; protocol specification".

[56] OMA-TS-PoC-System\_Description-V2\_1-20110802-A: "OMA PoC System Description".

[57] 3GPP TS 29.541: "5G System; Network Exposure (NE) function services for Non-IP Data Delivery (NIDD); Stage 3".

[58] 3GPP TS 29.522: "5G System; Network Exposure Function Northbound APIs; Stage 3".

[59] 3GPP TS 29.338: "Diameter based protocols to support Short Message Service (SMS) capable Mobile Management Entities (MMEs); Stage 3".

[60] 3GPP TS 29.337: "Diameter-based T4 interface for communications with packet data networks and applications".

[61] 3GPP TS 24.250: "Protocol for Reliable Data Service; Stage 3".

[62] 3GPP TS 29.128: "Mobility Management Entity (MME) and Serving GPRS Support Node (SGSN) interfaces for interworking with packet data networks and applications".

[63] 3GPP TS 29.122: "T8 reference point for Northbound APIs".

[64] 3GPP TS 29.598: "5G System; Unstructured Data Storage Services; Stage3".

[65] 3GPP TS 33.535: "Authentication and Key Management for Applications (AKMA) based on 3GPP credentials in the 5G System (5GS)".

[66] IETF RFC 5246: "The Transport Layer Security (TLS) Protocol Version 1.2".

[67] GSMA IR.88: "IR.88 LTE and EPC Roaming Guidelines".

[68] GSMA NG.114 "IMS Profile for Voice, Video and Messaging over 5GS".

[69] IETF RFC 8225: "PASSporT: Personal Assertion Token".

[70] IETF RFC 8224: "Authenticated Identity Management in the Session Initiation Protocol (SIP)".

[71] IETF RFC 8588: "Personal Assertion Token (PaSSporT) Extension for Signature-based Handling of Asserted information using toKENs (SHAKEN)".

[72] 3GPP TS 24.196: "Enhanced Calling Name (eCNAM)".

[73] IETF draft-ietf-stir-passport-rcd-17: "PASSporT Extension for Rich Call Data".

NOTE: The above document cannot be formally referenced until it is published as an RFC.

[74] 3GPP TS 24.229: "IP multimedia call control protocol based on Session Initiation Protocol (SIP)and Session Description Protocol (SDP); Stage 3".

[75] IANA Session Initiation Protocol (SIP) Parameters: <https://www.iana.org/assignments/sip-parameters/sip-parameters.xhtml>

[76] IETF RFC 8946: "Personal Assertion Token (PASSporT) Extension for Diverted Calls".

[77] 3GPP TS 23.204: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Support of Short Message Service (SMS) over generic 3GPP Internet Protocol (IP) access; Stage 2".

[78] GSMA RCC.07: "Rich Communication Suite – Advanced Communications Services and Client Specification".

[79] IETF RFC 4975: "The Message Session Relay Protocol (MSRP)".

[80] IETF RFC 3862: "Common Presence and Instant Messaging (CPIM): Message Format".

[81] IETF RFC 5438: "Instant Message Disposition Notification (IMDN)".

[82] OMA-TS-CPM\_System\_Description-V2\_2-20170926-C: "OMA Converged IP Messaging System Description".

[83] IETF RFC 4566: "SDP: Session Description Protocol".

[84] 3GPP TS 36.455: "Evolved Universal Terrestrial Radio Access (E-UTRA); LTE Positioning Protocol A (LPPa) ".

[85] 3GPP TS 37.355: "LTE Positioning Protocol (LPP)".

[86] 3GPP TS 38.455: "NG-RAN; NR Positioning Protocol A (NRPPa)".

[87] 3GPP TS 29.274: "3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3".

[88] 3GPP TS 29.513: "5G System; Policy and Charging Control signalling flows and QoS parameter mapping".

[89] 3GPP TS 29.512: "5G System; Session Management Policy Control Service; Stage 3".

[90] 3GPP TS 29.508: "5G System; Session Management Event Exposure Service; Stage 3".

[91] 3GPP TS 29.514: "5G System; Policy Authorization Service; Stage 3".

[92] 3GPP TS 29.214: "Policy and Charging Control over Rx reference point".

[93] 3GPP TS 24.558: "Enabling Edge Applications; Protocol specification".

[94] 3GPP TS 29.558: "Enabling Edge Applications; Application Programming Interface (API) specification".

[95] 3GPP TS 24.008: "Mobile radio interface Layer 3 specification; Core network protocols; Stage 3".

[96] 3GPP TS 29.551: "5G System; Packet Flow Description Management Service; Stage 3".

[97] ETSI TS 103 280: "Lawful Interception (LI); Dictionary for common parameters".

[98] 3GPP TS 26.512: "5G Media Streaming (5GMS); Protocols".

[99] 3GPP TS 26.247: "Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)".

[100] 3GPP TS 29.563: "5G System; Home Subscriber Server (HSS) services for interworking with Unified Data Management (UDM); Stage 3".

[101] 3GPP TS 29.562: "5G System; Home Subscriber Server (HSS) Services; Stage 3".

[102] 3GPP TS 24.341 "Support of SMS over IP networks, Stage 3".

[Re1] IETF RFC 2045: "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies".

[Re2] IETF RFC 5322: "Ineternet Message Format".

[Re3] IETF RFC 4975 : "The Message Session Relay Protocol (MSRP)".

## \*\*\*\* START OF NEXT CHANGE (MAIN DOCUMENT) \*\*\*

## 7.X Reuse of Externally Defined Structures

### 7.X.1 General

The subclauses below define LI structures to allow for the reuse of externally defined structures and schemas. These include:

- MIMEEntityContainer for sending information formated in a MIME Message.

- MSRPContainer for sending information formated in an MSRP Message.

If a message that is being reported cann carry multiple types of payloads described above, the following ReportedMessagePayload structure can be used.

Table 7.X.1-Ta1: Choices for ReportedMessagePayload

|  |  |  |
| --- | --- | --- |
| **Choice name** | **Type** | **Description** |
| mIMEEntityContainer | MIMEEntityContainer | Shall be chosen if the message contains a MIME Entity. |
| mSRPContainer | MSRPContainer | Shall be chosen if the message contains an MSRP Message. |

### 7.X.2 MIME message reporting

#### 7.X.2.1 General

When the payload of a message consists of a Multipurpose Internet Mail Extensions (MIME) Message as defined in IETF RFC 2045 [Re1], the following structures may be used to report that payload in IRI messages.

NOTE: As stated in IETF RFC 2045 [Re1], some of the definitions below are circular. This is unavoidable, since the overall structure of a MIME message is indeed recursive.

Depending on the requirements of the warrant and the contents of the message, it may be possible to encapsulate and send the entire MIME Message or MIME Entity. In this case, the encapuslatedMIMEEntity choice may be used as the value of the mIMEEntity parameter of the MIMEEntityContainer.

If the portions of the MIME Message or MIME Entity need to be removed, the separatedMIMEHeadersAndBody choice shall be used as the value of the mIMEEntity parameter of the MIMEEntityContainer.

#### 7.X.2.2 MIME message parameters

##### 7.X.2.2.1 Simple Types for MIME Messages

Table 7.X.2-Ta1: Simple Types for LI reporting of MIME Messages

|  |  |  |
| --- | --- | --- |
| Type name | Type definition | Description |
| MIMEContentType | UTF8String | Shall contain the contents of the Content-Type header field. See IETF RFC 2045 [Re1] clause 5. |
| MIMEHeaders | UTF8String | Shall contain the authorized MIME Headers. See IETF RFC 2045 [Re1] clause 3. |
| MIMEBody | UTF8String | Shall contain body of the MIME Entity. See IETF RFC 2045 [Re1] clauses 2.4 and 2.6. |
| MIMEHeaderBodyLength | INTEGER | Shall contain the length of the contents of the header field-body (see IETF RFC 5322 [Re2] clause 2.2) in bits. |
| MIMEHeaderFieldName | UTF8String | Shall contain the field-name of the header field (see IETF RFC 5322 [Re2] clause 2.2). |
| MIMEHeaderHash | OCTET STRING | Shall contain a hash of the header field-body. |
| MIMEBodyLength | INTEGER | Shall contain the length of the body of the MIME Entity in bits. |
| EncapsulatedMIMEEntity | OCTET STRING | Shall contain the entire MIME Entity. |

##### 7.X.2.2.2 Type: MIMEEntityContainer

Table 7.X.2-Ta2: Structure of the MIMEEntityContainer type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Type** | **Cardinality** | **Description** | **M/C/O** |
| contentType | MIMEContentType | 1 | Indicates the MIME content type of the Entity. | M |
| mIMEEntity | MIMEEntity | 1 | The contents of the MIME Entity. | M |

##### 7.X.2.2.3 Type: MIMEEntity

Table 7.X.2-Ta3: Choices for MIMEEntityType

|  |  |  |
| --- | --- | --- |
| **Choice name** | **Type** | **Description** |
| encapsulatedMIMEEntity | EncapsulatedMIMEEntity | The entire MIME Entity. |
| separatedMIMEHeadersAndBody | SeparatedMIMEHeadersAndBody | The MIME Entity separated into Headers and Body. |

##### 7.X.2.2.4 Type: SeparatedMIMEHeadersAndBody

Table 7.X.2-Ta4: Structure of the MIMEEntityContainer type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Type** | **Cardinality** | **Description** | **M/C/O** |
| entityHeaderSection | MIMEHeaderSection | 1 | The headers of the MIME Entity. | M |
| entityBody | MIMEEntityBody | 1 | The body of the MIME Entity. Shall be present if the Mime Entity has a body. | C |

##### 7.X.2.2.5 Type: MIMEHeaderSection

Table 7.X.2-Ta5: Structure of the MIMEHeaderSection type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Type** | **Cardinality** | **Description** | **M/C/O** |
| includedHeaderFields | MIMEHeaderFields | 1 | Shall contain the authorized headers of the entity. | M |
| removedHeaderFields | RemovedMIMEHeaderField | 0..MAX | Shall contain a sequence of information for any headers that were removed. | C |

##### 7.X.2.2.6 Type: MIMEEntityBody

Table 7.X.2-Ta6: Choices for MIMEEntityBody

|  |  |  |
| --- | --- | --- |
| **Choice name** | **Type** | **Description** |
| mIMEEntityContainer | MIMEEntityContainer | Shall be used if the body of the MIME Entity is another MIME Entity Container. This parameter shall contain the body of the MIME entity sent using the MIMEEntityContainer type. |
| bodyWithCommunicationsContent | MIMEEntityBodyWithCC | Shall be used if the body of the MIME Entity consists of only user generated communications content. |
| multipartBody | MultiPartMIMEBody | Shall be used if the body of the MIME Entity is a multipart body. |
| mSRPMessage | MSRPContainer | Shall be used if the body of the MIME Entity is an MSRP Message. |
| xMLBody | XMLType | Shall be used if the body of the MIME Entity is an XML Document. |
| sBIMessage | SBIType | Shall be used if the body of the MIME Entity is a 3GPP OpenAPI SBI Message. |
| body | MIMEBody | Shall be used if none of the other choices apply. This parameter shall contain the body of the MIME Entity as a UTF8 String. |

##### 7.X.2.2.7 Type: RemovedMIMEHeaderField

Table 7.X.2-Ta7: Structure of the RemovedMIMEHeaders type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Type** | **Cardinality** | **Description** | **M/C/O** |
| headerFieldName | MIMEHeaderFieldName | 1 | Shall be populated with the field name of the header that was removed. | M |
| headerBodyLength | MIMEHeaderBodyLength | 1 | The length of the removed header. | M |
| headerHash | MIMEHeaderHash | 0..1 | Hash of the removed header. | C |

##### 7.X.2.2.7 Type: MIMEEntityBodyWithCC

Table 7.X.2-Ta8: Choices for IMEEntityBodyWithCC

|  |  |  |
| --- | --- | --- |
| **Choice name** | **Type** | **Description** |
| mIMEBodyWithOnlyContent | MIMEBodyLength | The length of the MIME Body. Shall be chosen if all of the contents of the body of the MIME entity are communications content. |
| mIMEBodyWithXML | XMLContainer | Shall be chosen if the body of the MIME entity contains an XML document. |
| mIMEBodyWithSBI | SBIContainer | Shall be chosen if the body of the MIME entity contains a message in a format defined for the 5GC SBIs. |

### 7.X.3 MSRP Message reporting

#### 7.X.3.1 General

When the payload of a message consists of a Message Session Relay Protocol (MSRP) Message as defined in IETF RFC 4975 [Re3], the following structures may be used to report that payload in IRI messages.

Depending on the requirements of the warrant and the contents of the message, it may be possible to encapsulate and send the entire MSRP Message. In this case, the encapsulatedMSRP choice may be used as the value of the MSRPContainer.

If the portions of the MSRP Message or its payload need to be removed, the separatedMSRP choice shall be used as the value of the MSRPContainer.

#### 7.X.3.2 MSRP Message Parameters

##### 7.X.3.2.1 Simple Types for MSRP Messages

Table 7.X.3-Ta1: Simple Types for LI reporting of MSRP Messages

|  |  |  |
| --- | --- | --- |
| Type name | Type definition | Description |
| EncapsulatedMSRP | OCTET STRING | Shall contain the entire MSRP Message. |
| MSRPEndLine | UTF8String | Shall contain the value of the MSRP End Line. See IETF RFC 4975 [Re3] clause 7.1. |
| MSRPHeaders | UTF8String | Shall contain the MSRP Headers. See IETF RFC 4975 [Re3] clauses 7 and 15.2. |
| MSRPStartLine | UTF8String | Shall contain the value of the MSRP End Line. See IETF RFC 4975 [Re3] clause 7.1. |

##### 7.X.3.2.2 Type: MSRPContainer

Table 7.X.3-Ta2: Choices for MSRPContainer Type

|  |  |  |
| --- | --- | --- |
| **Choice name** | **Type** | **Description** |
| encapsulatedMSRP | EncapsulatedMSRP | The entire MSRP Message. |
| separatedMSRP | SeparatedMSRP | The MSRP message separated into Start Line, Headers, Body and End Line. |

##### 7.X.3.2.3 Type: SeparatedMSRP

Table 7.X.3-Ta3: Structure of the SeparatedMSRP type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Type** | **Cardinality** | **Description** | **M/C/O** |
| startLine | MSRPStartLine | 1 | The Start Line of the MSRP Message | M |
| headers | MSRPHeaders | 1 | The headers of the MSRP Message | M |
| body | MIMEEntityContainer | 0..1 | The body of the MSRP Message. Shall be present if the MSRP Message has a body. | C |
| endLine | MSRPEndLine | 1 | The End Line of the MSRP Message. | M |

## \*\*\*\* END OF MAIN DOCUMENTS CHANGES \*\*\*

## \*\*\*\* START OF FIRST CHANGE (ATTACHMENT TS33128Payloads.asn) \*\*\*

Annex A (normative):  
ASN.1 Schema for the Internal and External Interfaces

TS33128Payloads

{itu-t(0) identified-organization(4) etsi(0) securityDomain(2) lawfulIntercept(2) threeGPP(4) ts33128(19) r18(18) version2(2)}

DEFINITIONS IMPLICIT TAGS EXTENSIBILITY IMPLIED ::=

BEGIN

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

-- Relative OIDs

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

tS33128PayloadsOID RELATIVE-OID ::= {threeGPP(4) ts33128(19) r18(18) version2(2)}

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

{

-- AMF events, see clause 6.2.2.2

registration [1] AMFRegistration,

deregistration [2] AMFDeregistration,

locationUpdate [3] AMFLocationUpdate,

startOfInterceptionWithRegisteredUE [4] AMFStartOfInterceptionWithRegisteredUE,

unsuccessfulAMProcedure [5] AMFUnsuccessfulProcedure,

-- SMF events, see clause 6.2.3.2

pDUSessionEstablishment [6] SMFPDUSessionEstablishment,

pDUSessionModification [7] SMFPDUSessionModification,

pDUSessionRelease [8] SMFPDUSessionRelease,

startOfInterceptionWithEstablishedPDUSession [9] SMFStartOfInterceptionWithEstablishedPDUSession,

unsuccessfulSMProcedure [10] SMFUnsuccessfulProcedure,

-- UDM events, see clause 7.2.2.3

servingSystemMessage [11] UDMServingSystemMessage,

-- SMS events, see clause 6.2.5.2

sMSMessage [12] SMSMessage,

-- LALS events, see clause 7.3.1.4

lALSReport [13] LALSReport,

-- PDHR/PDSR events, see clauses 6.2.3.5 and 6.2.3.9

pDHeaderReport [14] PDHeaderReport,

pDSummaryReport [15] PDSummaryReport,

-- Tag 16 is reserved because there is no equivalent mDFCellSiteReport in XIRIEvent.

-- MMS events, see clause 7.4.3

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 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,

-- UDM events, see clause 7.2.2.3, continued from tag 11

subscriberRecordChangeMessage [54] UDMSubscriberRecordChangeMessage,

cancelLocationMessage [55] UDMCancelLocationMessage,

-- SMS events, see clause 6.2.5.2, continued from tag 12

sMSReport [56] SMSReport,

-- SMF MA PDU session 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,

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

sMFPDUtoMAPDUSessionModification [64] SMFPDUtoMAPDUSessionModification,

-- NEF events, see clause 7.7.2.1

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 events, see clause 7.8.2.1

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,

-- 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 clauses 7.9.1.3 and 7.9.1.4

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 7.3.4.1

separatedLocationReporting [102] SeparatedLocationReporting,

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

sTIRSHAKENSignatureGeneration [103] STIRSHAKENSignatureGeneration,

sTIRSHAKENSignatureValidation [104] STIRSHAKENSignatureValidation,

-- IMS events, see clause 7.12.4.2

iMSMessage [105] IMSMessage,

startOfInterceptionForActiveIMSSession [106] StartOfInterceptionForActiveIMSSession,

iMSCCUnavailable [107] IMSCCUnavailable,

-- UDM events, see clause 7.2.2.3, continued from tag 55

uDMLocationInformationResult [108] UDMLocationInformationResult,

uDMUEInformationResponse [109] UDMUEInformationResponse,

uDMUEAuthenticationResponse [110] UDMUEAuthenticationResponse,

-- AMF events, see 6.2.2.2.8, continued from tag 5

positioningInfoTransfer [111] AMFPositioningInfoTransfer,

-- MME events, see clause 6.3.2.2.8, continued from tag 91

mMEPositioningInfoTransfer [112] MMEPositioningInfoTransfer,

-- AMF events, see 6.2.2.2.9, continued from tag 111

aMFRANHandoverCommand [113] AMFRANHandoverCommand,

aMFRANHandoverRequest [114] AMFRANHandoverRequest,

-- EES events, see clause 7.14.2

eESEECRegistration [115] EESEECRegistration,

eESEASDiscovery [116] EESEASDiscovery,

eESEASDiscoverySubscription [117] EESEASDiscoverySubscription,

eESEASDiscoveryNotification [118] EESEASDiscoveryNotification,

eESAppContextRelocation [119] EESAppContextRelocation,

eESACRSubscription [120] EESACRSubscription,

eESACRNotification [121] EESACRNotification,

eESEECContextRelocation [122] EESEECContextRelocation,

eESStartOfInterceptionWithRegisteredEEC [123] EESStartOfInterceptionWithRegisteredEEC,

-- UDM events, see clause 7.2.2.3, continued from tag 110

uDMStartOfInterceptionWithRegisteredTarget [124] UDMStartOfInterceptionWithRegisteredTarget,

-- 5GMS AF events, see clause 7.15.2

fiveGMSAFServiceAccessInformation [125] FiveGMSAFServiceAccessInformation,

fiveGMSAFConsumptionReporting [126] FiveGMSAFConsumptionReporting,

fiveGMSAFDynamicPolicyInvocation [127] FiveGMSAFDynamicPolicyInvocation,

fiveGMSAFMetricsReporting [128] FiveGMSAFMetricsReporting,

fiveGMSAFNetworkAssistance [129] FiveGMSAFNetworkAssistance,

fiveGMSAFUnsuccessfulProcedure [130] FiveGMSAFUnsuccessfulProcedure,

fiveGMSAFStartOfInterceptionWithAlreadyConfiguredUE [131] FiveGMSAFStartOfInterceptionWithAlreadyConfiguredUE,

--AMF events, see 6.2.2.2.10, continued from tag 114

aMFUEConfigurationUpdate [132] AMFUEConfigurationUpdate,

-- HSS events, see clause 7.2.3.3

hSSServingSystemMessage [133] HSSServingSystemMessage,

hSSStartOfInterceptionWithRegisteredTarget [134] HSSStartOfInterceptionWithRegisteredTarget

}

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

-- 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

{

-- AMF events, see clause 6.2.2.3

registration [1] AMFRegistration,

deregistration [2] AMFDeregistration,

locationUpdate [3] AMFLocationUpdate,

startOfInterceptionWithRegisteredUE [4] AMFStartOfInterceptionWithRegisteredUE,

unsuccessfulAMProcedure [5] AMFUnsuccessfulProcedure,

-- SMF events, see clause 6.2.3.7

pDUSessionEstablishment [6] SMFPDUSessionEstablishment,

pDUSessionModification [7] SMFPDUSessionModification,

pDUSessionRelease [8] SMFPDUSessionRelease,

startOfInterceptionWithEstablishedPDUSession [9] SMFStartOfInterceptionWithEstablishedPDUSession,

unsuccessfulSMProcedure [10] SMFUnsuccessfulProcedure,

-- UDM events, see clause 7.2.2.4

servingSystemMessage [11] UDMServingSystemMessage,

-- SMS events, see clause 6.2.5.4

sMSMessage [12] SMSMessage,

-- LALS events, see clause 7.3.1.5

lALSReport [13] LALSReport,

-- PDHR/PDSR events, see clause 6.2.3.9

pDHeaderReport [14] PDHeaderReport,

pDSummaryReport [15] PDSummaryReport,

-- MDF events, see clause 7.3.2.2

mDFCellSiteReport [16] MDFCellSiteReport,

-- MMS events, see clause 7.4.4.1

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 events, see clauses 7.5.2 and 7.5.3.1

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,

-- UDM events, see clause 7.2.2.4, continued from tag 11

subscriberRecordChangeMessage [54] UDMSubscriberRecordChangeMessage,

cancelLocationMessage [55] UDMCancelLocationMessage,

-- SMS events, see clause 6.2.5.4, continued from tag 12

sMSReport [56] SMSReport,

-- SMF MA PDU session events, see clause 6.2.3.7

sMFMAPDUSessionEstablishment [57] SMFMAPDUSessionEstablishment,

sMFMAPDUSessionModification [58] SMFMAPDUSessionModification,

sMFMAPDUSessionRelease [59] SMFMAPDUSessionRelease,

startOfInterceptionWithEstablishedMAPDUSession [60] SMFStartOfInterceptionWithEstablishedMAPDUSession,

unsuccessfulMASMProcedure [61] SMFMAUnsuccessfulProcedure,

-- Identifier Association events, see clauses 6.2.2.3 and 6.3.2.3

aMFIdentifierAssociation [62] AMFIdentifierAssociation,

mMEIdentifierAssociation [63] MMEIdentifierAssociation,

-- SMF PDU to MA PDU session events, see clause 6.2.3.7

sMFPDUtoMAPDUSessionModification [64] SMFPDUtoMAPDUSessionModification,

-- NEF events, see clause 7.7.2.3

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 events, see clause 7.8.2.3

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,

-- MME events, see clause 6.3.2.3

mMEAttach [87] MMEAttach,

mMEDetach [88] MMEDetach,

mMELocationUpdate [89] MMELocationUpdate,

mMEStartOfInterceptionWithEPSAttachedUE [90] MMEStartOfInterceptionWithEPSAttachedUE,

mMEUnsuccessfulProcedure [91] MMEUnsuccessfulProcedure,

-- AKMA key management events, see clause 7.9.1.5

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 7.3.4.1

separatedLocationReporting [102] SeparatedLocationReporting,

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

sTIRSHAKENSignatureGeneration [103] STIRSHAKENSignatureGeneration,

sTIRSHAKENSignatureValidation [104] STIRSHAKENSignatureValidation,

-- IMS events, see clause 7.12.7

iMSMessage [105] IMSMessage,

startOfInterceptionForActiveIMSSession [106] StartOfInterceptionForActiveIMSSession,

iMSCCUnavailable [107] IMSCCUnavailable,

-- UDM events, see clause 7.2.2.4, continued from tag 55

uDMLocationInformationResult [108] UDMLocationInformationResult,

uDMUEInformationResponse [109] UDMUEInformationResponse,

uDMUEAuthenticationResponse [110] UDMUEAuthenticationResponse,

-- AMF events, see 6.2.2.3, continued from tag 5

positioningInfoTransfer [111] AMFPositioningInfoTransfer,

-- MME events, see clause 6.3.2.3, continued from tag 91

mMEPositioningInfoTransfer [112] MMEPositioningInfoTransfer,

-- AMF events, see 6.2.2.3, continued from tag 111

aMFRANHandoverCommand [113] AMFRANHandoverCommand,

aMFRANHandoverRequest [114] AMFRANHandoverRequest,

-- EES events, see clause 7.14.2.11

eESEECRegistration [115] EESEECRegistration,

eESEASDiscovery [116] EESEASDiscovery,

eESEASDiscoverySubscription [117] EESEASDiscoverySubscription,

eESEASDiscoveryNotification [118] EESEASDiscoveryNotification,

eESAppContextRelocation [119] EESAppContextRelocation,

eESACRSubscription [120] EESACRSubscription,

eESACRNotification [121] EESACRNotification,

eESEECContextRelocation [122] EESEECContextRelocation,

eESStartOfInterceptionWithRegisteredEEC [123] EESStartOfInterceptionWithRegisteredEEC,

-- UDM events, see clause 7.2.2.4, continued from tag 110

uDMStartOfInterceptionWithRegisteredTarget [124] UDMStartOfInterceptionWithRegisteredTarget,

-- 5GMS AF events, see clause 7.15.3

fiveGMSAFServiceAccessInformation [125] FiveGMSAFServiceAccessInformation,

fiveGMSAFConsumptionReporting [126] FiveGMSAFConsumptionReporting,

fiveGMSAFDynamicPolicyInvocation [127] FiveGMSAFDynamicPolicyInvocation,

fiveGMSAFMetricsReporting [128] FiveGMSAFMetricsReporting,

fiveGMSAFNetworkAssistance [129] FiveGMSAFNetworkAssistance,

fiveGMSAFUnsuccessfulProcedure [130] FiveGMSAFUnsuccessfulProcedure,

fiveGMSAFStartOfInterceptionWithAlreadyConfiguredUE [131] FiveGMSAFStartOfInterceptionWithAlreadyConfiguredUE,

--AMF events, see 6.2.2.3, continued from tag 114

aMFUEConfigurationUpdate [132] AMFUEConfigurationUpdate,

-- HSS events, see clause 7.2.3.4

hSSServingSystemMessage [133] HSSServingSystemMessage,

hSSStartOfInterceptionWithRegisteredTarget [134] HSSStartOfInterceptionWithRegisteredTarget

}

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,

-- In Rel-16 (threeGPP(4) ts33128(19) r16(16) version9(9)),

-- tag 4 is pTCCCPDU and tag 5 is not used.

-- Rel-17 or newer decoders should decode tag 4 contents as PTCCCPDU if

-- r16 is used in cCPayloadOID.

nIDDCCPDU [4] NIDDCCPDU,

pTCCCPDU [5] PTCCCPDU,

iMSCCPDU [6] IMSCCPDU

}

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

-- 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,

pagingRestrictionIndicator [17] PagingRestrictionIndicator 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))

}

-- See clause 6.2.2.2.9.2 for details of this structure

AMFRANHandoverCommand ::= SEQUENCE

{

userIdentifiers [1] UserIdentifiers,

aMFUENGAPID [2] AMFUENGAPID,

rANUENGAPID [3] RANUENGAPID,

handoverType [4] HandoverType,

targetToSourceContainer [5] RANTargetToSourceContainer

}

-- See clause 6.2.2.2.9.3 for details of this structure

AMFRANHandoverRequest ::= SEQUENCE

{

userIdentifiers [1] UserIdentifiers,

aMFUENGAPID [2] AMFUENGAPID,

rANUENGAPID [3] RANUENGAPID,

handoverType [4] HandoverType,

handoverCause [5] HandoverCause,

pDUSessionResourceInformation [6] PDUSessionResourceInformation,

mobilityRestrictionList [7] MobilityRestrictionList OPTIONAL,

locationReportingRequestType [8] LocationReportingRequestType OPTIONAL,

targetToSourceContainer [9] RANTargetToSourceContainer,

nPNAccessInformation [10] NPNAccessInformation OPTIONAL,

sourceToTargetContainer [11] RANSourceToTargetContainer

}

--See clause 6.2.2.2.10 on for details of this structure

AMFUEConfigurationUpdate ::= SEQUENCE

{

userIdentifiers [1] UserIdentifiers,

gUTI [2] GUTI,

oldGUTI [3] EPS5GGUTI OPTIONAL,

fiveGSTAIList [4] TAIList OPTIONAL,

slice [5] Slice OPTIONAL,

serviceAreaList [6] ServiceAreaList OPTIONAL,

registrationResult [7] AMFRegistrationResult OPTIONAL,

sMSOverNASIndicator [8] SMSOverNASIndicator OPTIONAL

}

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

-- 5G AMF parameters

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

AMFID ::= SEQUENCE

{

aMFRegionID [1] AMFRegionID,

aMFSetID [2] AMFSetID,

aMFPointer [3] AMFPointer

}

AMFDirection ::= ENUMERATED

{

networkInitiated(1),

uEInitiated(2)

}

AMFFailedProcedureType ::= ENUMERATED

{

registration(1),

sMS(2),

pDUSessionEstablishment(3)

}

AMFFailureCause ::= CHOICE

{

fiveGMMCause [1] FiveGMMCause,

fiveGSMCause [2] FiveGSMCause

}

AMFPointer ::= INTEGER (0..63)

AMFRegistrationResult ::= ENUMERATED

{

threeGPPAccess(1),

nonThreeGPPAccess(2),

threeGPPAndNonThreeGPPAccess(3)

}

AMFRegionID ::= INTEGER (0..255)

AMFRegistrationType ::= ENUMERATED

{

initial(1),

mobility(2),

periodic(3),

emergency(4),

sNPNOnboarding(5),

disasterMobility(6),

disasterInitial(7)

}

AMFSetID ::= INTEGER (0..1023)

AMFUENGAPID ::= INTEGER (0..1099511627775)

-- TS 24.501 [13], clause 9.11.3.49

ServiceAreaList ::= OCTET STRING (SIZE(4..112))

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

-- 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,

selectedDNN [21] DNN OPTIONAL,

servingNetwork [22] SMFServingNetwork OPTIONAL,

oldPDUSessionID [23] PDUSessionID OPTIONAL,

handoverState [24] HandoverState OPTIONAL,

gTPTunnelInfo [25] GTPTunnelInfo OPTIONAL,

pCCRules [26] PCCRuleSet OPTIONAL,

ePSPDNConnectionEstablishment [27] 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,

uEEndpoint [13] UEEndpointAddress OPTIONAL,

servingNetwork [14] SMFServingNetwork OPTIONAL,

handoverState [15] HandoverState OPTIONAL,

gTPTunnelInfo [16] GTPTunnelInfo OPTIONAL,

pCCRules [17] PCCRuleSet OPTIONAL,

ePSPDNConnectionModification[18] EPSPDNConnectionModification OPTIONAL,

uPPathChange [19] UPPathChange OPTIONAL,

pFDDataForApp [20] PFDDataForApp 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,

nGAPCause [12] NGAPCauseInt OPTIONAL,

fiveGMMCause [13] FiveGMMCause OPTIONAL,

pCCRuleIDs [14] PCCRuleIDSet OPTIONAL,

ePSPDNConnectionRelease [15] EPSPDNConnectionRelease 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,

uEEPSPDNConnection [21] UEEPSPDNConnection OPTIONAL,

servingNetwork [22] SMFServingNetwork OPTIONAL,

gTPTunnelInfo [23] GTPTunnelInfo OPTIONAL,

pCCRules [24] PCCRuleSet OPTIONAL,

ePSStartOfInterceptionWithEstablishedPDNConnection [25] EPSStartOfInterceptionWithEstablishedPDNConnection OPTIONAL,

pFDDataForApps [26] PFDDataForApps 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,

uEEndpoint [14] UEEndpointAddress OPTIONAL,

servingNetwork [15] SMFServingNetwork OPTIONAL,

handoverState [16] HandoverState OPTIONAL,

gTPTunnelInfo [17] GTPTunnelInfo OPTIONAL,

ePSPDNConnectionModification [18] EPSPDNConnectionModification OPTIONAL

}

-- 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,

uEEPSPDNConnection [22] UEEPSPDNConnection OPTIONAL,

ePS5GSComboInfo [23] EPS5GSComboInfo OPTIONAL,

selectedDNN [24] DNN OPTIONAL,

handoverState [25] HandoverState OPTIONAL,

pCCRules [26] PCCRuleSet OPTIONAL,

ePSPDNConnectionEstablishment [27] EPSPDNConnectionEstablishment 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,

uEEPSPDNConnection [16] UEEPSPDNConnection OPTIONAL,

ePS5GSComboInfo [17] EPS5GSComboInfo OPTIONAL,

handoverState [18] HandoverState OPTIONAL,

pCCRules [19] PCCRuleSet OPTIONAL,

uPPathChange [20] UPPathChange OPTIONAL,

pFDDataForApp [21] PFDDataForApp OPTIONAL,

ePSPDNConnectionModification [22] EPSPDNConnectionModification 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,

nGAPCause [11] NGAPCauseInt OPTIONAL,

fiveGMMCause [12] FiveGMMCause OPTIONAL,

pCCRuleIDs [13] PCCRuleIDSet OPTIONAL,

ePSPDNConnectionRelease [14] EPSPDNConnectionRelease 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,

ePS5GSComboInfo [22] EPS5GSComboInfo OPTIONAL,

uEEPSPDNConnection [23] UEEPSPDNConnection OPTIONAL,

pCCRules [24] PCCRuleSet OPTIONAL,

pFDDataForApps [25] PFDDataForApps OPTIONAL,

ePSStartOfInterceptionWithEstablishedPDNConnection [26] EPSStartOfInterceptionWithEstablishedPDNConnection 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,

gTPTunnelInfo [7] GTPTunnelInfo OPTIONAL

}

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

ATSSSContainer ::= OCTET STRING

DLRANTunnelInformation ::= SEQUENCE

{

dLQOSFlowTunnelInformation [1] QOSFlowTunnelInformation OPTIONAL,

additionalDLQOSFlowTunnelInformation [2] QOSFlowTunnelInformationList OPTIONAL,

redundantDLQOSFlowTunnelInformation [3] QOSFlowTunnelInformationList OPTIONAL,

additionalredundantDLQOSFlowTunnelInformation [4] QOSFlowTunnelInformationList OPTIONAL

}

EstablishmentStatus ::= ENUMERATED

{

established(0),

released(1)

}

FiveGSGTPTunnels ::= SEQUENCE

{

uLNGUUPTunnelInformation [1] FTEID OPTIONAL,

additionalULNGUUPTunnelInformation [2] FTEIDList OPTIONAL,

dLRANTunnelInformation [3] DLRANTunnelInformation OPTIONAL

}

FiveQI ::= INTEGER (0..255)

HandoverState ::= ENUMERATED

{

none(1),

preparing(2),

prepared(3),

completed(4),

cancelled(5)

}

NGAPCauseInt ::= SEQUENCE

{

group [1] NGAPCauseGroupInt,

value [2] NGAPCauseValueInt

}

-- Derived as described in TS 29.571 [17] clause 5.4.4.12

NGAPCauseGroupInt ::= INTEGER

NGAPCauseValueInt ::= INTEGER

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)

}

QOSFlowTunnelInformation ::= SEQUENCE

{

uPTunnelInformation [1] FTEID,

associatedQOSFlowList [2] QOSFlowLists

}

QOSFlowTunnelInformationList ::= SEQUENCE OF QOSFlowTunnelInformation

QOSFlowDescription ::= OCTET STRING

QOSFlowLists ::= SEQUENCE OF QOSFlowList

QOSFlowList ::= SEQUENCE

{

qFI [1] QFI,

qOSRules [2] QOSRules OPTIONAL,

eBI [3] EPSBearerID OPTIONAL,

qOSFlowDescription [4] QOSFlowDescription OPTIONAL,

qOSFlowProfile [5] QOSFlowProfile OPTIONAL,

associatedANType [6] AccessType OPTIONAL,

defaultQOSRuleIndication [7] BOOLEAN OPTIONAL

}

QOSFlowProfile ::= SEQUENCE

{

fiveQI [1] FiveQI

}

QOSRules ::= OCTET STRING

-- See clauses 5.6.2.6-1 and 5.6.2.9-1 of TS 29.512 [89], clause table 5.6.2.5-1 of TS 29.508 [90] for the details of this structure

PCCRule ::= SEQUENCE

{

pCCRuleID [1] PCCRuleID OPTIONAL,

appId [2] UTF8String OPTIONAL,

flowInfos [3] FlowInformationSet OPTIONAL,

appReloc [4] BOOLEAN OPTIONAL,

simConnInd [5] BOOLEAN OPTIONAL,

simConnTerm [6] INTEGER OPTIONAL,

maxAllowedUpLat [7] INTEGER OPTIONAL,

trafficRoutes [8] RouteToLocationSet,

trafficSteeringPolIdDl [9] UTF8String OPTIONAL,

trafficSteeringPolIdUl [10] UTF8String OPTIONAL,

sourceDNAI [11] DNAI OPTIONAL,

targetDNAI [12] DNAI OPTIONAL,

dNAIChangeType [13] DNAIChangeType OPTIONAL,

sourceUEIPAddr [14] IPAddress OPTIONAL,

targetUEIPAddr [15] IPAddress OPTIONAL,

sourceTrafficRouting [16] RouteToLocation OPTIONAL,

targetTrafficRouting [17] RouteToLocation OPTIONAL,

eASIPReplaceInfos [18] EASIPReplaceInfos OPTIONAL

}

-- See clause table 5.6.2.5-1 of TS 29.508 [90] for the details of this structure.

UPPathChange ::= SEQUENCE

{

sourceDNAI [1] DNAI OPTIONAL,

targetDNAI [2] DNAI OPTIONAL,

dNAIChangeType [3] DNAIChangeType OPTIONAL,

sourceUEIPAddr [4] IPAddress OPTIONAL,

targetUEIPAddr [5] IPAddress OPTIONAL,

sourceTrafficRouting [6] RouteToLocation OPTIONAL,

targetTrafficRouting [7] RouteToLocation OPTIONAL,

mACAddress [8] MACAddress OPTIONAL

}

-- See table 5.6.2.14-1 of TS 29.512 [89]

PCCRuleID ::= UTF8String

PCCRuleSet ::= SET OF PCCRule

PCCRuleIDSet ::= SET OF PCCRuleID

FlowInformationSet ::= SET OF FlowInformation

RouteToLocationSet ::= SET OF RouteToLocation

-- See table 5.6.2.14 of TS 29.512 [89]

FlowInformation ::= SEQUENCE

{

flowDescription [1] FlowDescription OPTIONAL,

ethFlowDescription [2] EthFlowDescription OPTIONAL,

tosTrafficClass [3] OCTET STRING (SIZE(2)) OPTIONAL,

spi [4] OCTET STRING (SIZE(4)) OPTIONAL,

flowLabel [5] OCTET STRING (SIZE(3)) OPTIONAL,

flowDirection [6] FlowDirection OPTIONAL

}

-- See table 5.6.2.14 of TS 29.512 [89]

FlowDescription ::= SEQUENCE

{

sourceIPAddress [1] IPAddressOrRangeOrAny,

destinationIPAddress [2] IPAddressOrRangeOrAny,

sourcePortNumber [3] PortNumber OPTIONAL,

destinationPortNumber [4] PortNumber OPTIONAL,

protocol [5] NextLayerProtocolOrAny

}

IPAddressOrRangeOrAny ::= CHOICE

{

iPAddress [1] IPAddress,

ipAddressRange [2] IPMask,

anyIPAddress [3] AnyIPAddress

}

IPMask ::= SEQUENCE

{

fromIPAddress [1] IPAddress,

toIPAddress [2] IPAddress

}

AnyIPAddress ::= ENUMERATED

{

any(1)

}

NextLayerProtocolOrAny ::= CHOICE

{

nextLayerProtocol [1] NextLayerProtocol,

anyNextLayerProtocol [2] AnyNextLayerProtocol

}

AnyNextLayerProtocol ::= ENUMERATED

{

ip(1)

}

-- See table 5.6.2.17-1 of TS 29.514 [91]

EthFlowDescription ::= SEQUENCE

{

destMacAddress [1] MACAddress OPTIONAL,

ethType [2] OCTET STRING (SIZE(2)),

fDesc [3] FlowDescription OPTIONAL,

fDir [4] FDir OPTIONAL,

sourceMacAddress [5] MACAddress OPTIONAL,

vlanTags [6] SET OF VLANTag,

srcMacAddrEnd [7] MACAddress OPTIONAL,

destMacAddrEnd [8] MACAddress OPTIONAL

}

-- See table 5.6.2.17-1 of TS 29.514 [91]

FDir ::= ENUMERATED

{

downlink(1)

}

-- See table 5.6.2.17-1 of TS 29.514 [91]

VLANTag ::= SEQUENCE

{

priority [1] BIT STRING (SIZE(3)),

cFI [2] BIT STRING (SIZE(1)),

vLANID [3] BIT STRING (SIZE(12))

}

-- See table 5.6.2.14 of TS 29.512 [89]

FlowDirection ::= ENUMERATED

{

downlinkOnly(1),

uplinkOnly(2),

dowlinkAndUplink(3)

}

-- See table 5.4.2.1 of TS 29.571 [17]

DNAIChangeType ::= ENUMERATED

{

early(1),

earlyAndLate(2),

late(3)

}

-- See table 5.6.2.15 of TS 29.571 [17]

RouteToLocation ::= SEQUENCE

{

dNAI [1] DNAI,

routeInfo [2] RouteInfo

}

-- See table 5.4.2.1 of TS 29.571 [17]

DNAI ::= UTF8String

-- See table 5.4.4.16 of TS 29.571 [17]

RouteInfo ::= SEQUENCE

{

iPAddressTunnelEndpoint [1] IPAddress,

uDPPortNumberTunnelEndpoint [2] PortNumber

}

-- See clause 4.1.4.2 of TS 29.512 [89]

EASIPReplaceInfos ::= SEQUENCE

{

sourceEASAddress [1] EASServerAddress,

targetEASAddress [2] EASServerAddress

}

-- See clause 4.1.4.2 of TS 29.512 [89]

EASServerAddress ::= SEQUENCE

{

iPAddress [1] IPAddress,

port [2] PortNumber

}

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

-- PGW-C + SMF PDNConnection Events

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

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

}

EPSPDNConnectionModification ::= 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 OPTIONAL,

bearerConcextsModified [18] SEQUENCE OF EPSBearerContextModified,

bearerContextsMarkedForRemoval [19] SEQUENCE OF EPSBearerContextForRemoval OPTIONAL,

bearersDeleted [20] SEQUENCE OF EPSBearersDeleted OPTIONAL,

indicationFlags [21] PDNConnectionIndicationFlags OPTIONAL,

handoverIndication [22] PDNHandoverIndication OPTIONAL,

nBIFOMSupport [23] PDNNBIFOMSupport OPTIONAL,

fiveGSInterworkingInfo [24] FiveGSInterworkingInfo OPTIONAL,

cSRMFI [25] CSRMFI OPTIONAL,

restorationOfPDNConnectionsSupport [26] RestorationOfPDNConnectionsSupport OPTIONAL,

pGWChangeIndication [27] PGWChangeIndication OPTIONAL,

pGWRNSI [28] PGWRNSI OPTIONAL

}

EPSPDNConnectionRelease ::= SEQUENCE

{

ePSSubscriberIDs [1] EPSSubscriberIDs,

iMSIUnauthenticated [2] IMSIUnauthenticatedIndication OPTIONAL,

defaultBearerID [3] EPSBearerID,

location [4] Location OPTIONAL,

gTPTunnelInfo [5] GTPTunnelInfo OPTIONAL,

rANNASCause [6] EPSRANNASCause OPTIONAL,

pDNConnectionType [7] PDNConnectionType,

indicationFlags [8] PDNConnectionIndicationFlags OPTIONAL,

scopeIndication [9] EPSPDNConnectionReleaseScopeIndication OPTIONAL,

bearersDeleted [10] SEQUENCE OF EPSBearersDeleted OPTIONAL

}

EPSStartOfInterceptionWithEstablishedPDNConnection ::= 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,

bearerContexts [17] SEQUENCE OF EPSBearerContext

}

PFDDataForApps ::= SET OF PFDDataForApp

PFDDataForApp ::= SEQUENCE

{

aPPId [1] UTF8String,

pFDs [2] PFDs

}

PFDs ::= SET OF PFD

-- See clause 5.6.2.5 of TS 29.551 [96]

PFD ::= SEQUENCE

{

pFDId [1] UTF8String,

pFDFlowDescriptions [2] PFDFlowDescriptions,

urls [3] PFDURLs,

domainNames [4] DomainNames,

dnProtocol [5] DnProtocol

}

PFDURLs ::= SET OF UTF8String

PFDFlowDescriptions ::= SET OF PFDFlowDescription

DomainNames ::= SET OF UTF8String

PFDFlowDescription ::= SEQUENCE

{

nextLayerProtocol [1] NextLayerProtocol,

serverIPAddress [2] IPAddress,

serverPortNumber [3] PortNumber

}

-- See clause 5.14.2.2.4 of TS 29.122 [63]

DnProtocol ::= ENUMERATED

{

dnsQname(1),

tlsSni(2),

tlsSan(3),

tlsScn(4)

}

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

-- 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

}

EPSBearerInfo ::= SEQUENCE OF EPSBearers

EPSBearers ::= SEQUENCE

{

ePSBearerID [1] EPSBearerID,

pGWS8UserPlaneFTEID [2] FTEID,

qCI [3] QCI

}

EPSBearerContext ::= SEQUENCE

{

ePSBearerID [1] EPSBearerID,

uPGTPTunnelInfo [2] GTPTunnelInfo,

bearerQOS [3] EPSBearerQOS

}

EPSBearerContextCreated ::= SEQUENCE

{

ePSBearerID [1] EPSBearerID,

cause [2] EPSBearerCreationCauseValue,

gTPTunnelInfo [3] GTPTunnelInfo OPTIONAL,

bearerQOS [4] EPSBearerQOS OPTIONAL,

protocolConfigurationOptions [5] PDNProtocolConfigurationOptions OPTIONAL

}

EPSBearerContextModified ::= SEQUENCE

{

ePSBearerID [1] EPSBearerID,

cause [2] EPSBearerModificationCauseValue,

gTPTunnelInfo [3] GTPTunnelInfo OPTIONAL,

bearerQOS [4] EPSBearerQOS OPTIONAL,

protocolConfigurationOptions [5] PDNProtocolConfigurationOptions OPTIONAL

}

EPSBearersDeleted ::= SEQUENCE

{

linkedEPSBearerID [1] EPSBearerID OPTIONAL,

ePSBearerIDs [2] SEQUENCE OF EPSBearerID OPTIONAL,

protocolConfigurationOptions [3] PDNProtocolConfigurationOptions OPTIONAL,

cause [4] EPSBearerDeletionCauseValue OPTIONAL,

deleteBearerResponse [5] EPSDeleteBearerResponse

}

EPSDeleteBearerResponse ::= SEQUENCE

{

cause [1] EPSBearerDeletionCauseValue,

linkedEPSBearerID [2] EPSBearerID OPTIONAL,

bearerContexts [3] SEQUENCE OF EPSDeleteBearerContext OPTIONAL,

protocolConfigurationOptions [4] PDNProtocolConfigurationOptions OPTIONAL

}

EPSDeleteBearerContext ::= SEQUENCE

{

cause [1] EPSBearerDeletionCauseValue,

ePSBearerID [2] EPSBearerID,

protocolConfigurationOptions [3] PDNProtocolConfigurationOptions OPTIONAL,

rANNASCause [4] EPSRANNASCause OPTIONAL

}

EPSBearerContextForRemoval ::= SEQUENCE

{

ePSBearerID [1] EPSBearerID,

cause [2] EPSBearerRemovalCauseValue

}

EPSBearerCreationCauseValue ::= INTEGER (0..255)

EPSBearerDeletionCauseValue ::= INTEGER (0..255)

EPSBearerModificationCauseValue ::= INTEGER (0..255)

EPSBearerRemovalCauseValue ::= INTEGER (0..255)

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

}

EPSRANNASCause ::= OCTET STRING

EPSQOSPriority ::= INTEGER (1..15)

BitrateBinKBPS ::= OCTET STRING

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)

}

EPSPDNConnectionReleaseScopeIndication ::= BOOLEAN

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)

GTPTunnelInfo ::= SEQUENCE

{

fiveGSGTPTunnels [1] FiveGSGTPTunnels OPTIONAL,

ePSGTPTunnels [2] EPSGTPTunnels OPTIONAL

}

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,

roamingIndicator [9] RoamingIndicator 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,

aMFDeregistrationInfo [7] UDMAMFDeregistrationInfo OPTIONAL,

deregistrationData [8] UDMDeregistrationData OPTIONAL

}

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

}

UDMStartOfInterceptionWithRegisteredTarget ::= SEQUENCE

{

sUPI [1] SUPI,

gPSI [2] GPSI OPTIONAL,

uDMSubscriptionDataSets [3] SBIType

}

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

-- 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

}

RoamingIndicator ::= BOOLEAN

UDMAMFDeregistrationInfo ::= SEQUENCE

{

gUAMI [1] GUAMI,

purgeFlag [2] BOOLEAN

}

UDMDeregistrationData ::= SEQUENCE

{

deregReason [1] UDMDeregReason OPTIONAL,

accessType [2] AccessType OPTIONAL,

pDUSessionID [3] PDUSessionID OPTIONAL

}

UDMDeregReason ::= ENUMERATED

{

uEInitialRegistration(1),

uERegistrationAreaChange(2),

subscriptionWithdrawn(3),

fiveGSToEPSMobility(4),

fiveGSToEPSMobilityUeInitialRegistration(5),

reregistrationRequired(6),

sMFContextTransferred(7),

duplicatePDUSession(8),

fiveGSRVCCToUTRANMobility(9)

}

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

-- 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 CCPDU

-- =========

IMSCCPDU ::= SEQUENCE

{

payload [1] IMSCCPDUPayload,

sDPInfo [2] OCTET STRING OPTIONAL

}

IMSCCPDUPayload ::= OCTET STRING

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

-- IMS parameters

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

IMSPayload ::= CHOICE

{

encapsulatedSIPMessage [1] SIPMessage,

separatedSIPHeaderAndBody [2] SeparatedSIPHeaderAndBody

}

SIPMessage ::= SEQUENCE

{

iPSourceAddress [1] IPAddress,

iPDestinationAddress [2] IPAddress,

sIPContent [3] OCTET STRING

}

SeparatedSIPHeaderAndBody ::= SEQUENCE

{

iPSourceAddress [1] IPAddress,

iPDestinationAddress [2] IPAddress,

separatedSIPContent [3] MIMEEntityContainer

}

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,

separatedSIPHeaderAndBody [3] SeparatedSIPHeaderAndBody 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,

separatedSIPHeaderAndBody [7] SeparatedSIPHeaderAndBody 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

}

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

-- EES definitions

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

-- See clause 7.14.2.2 for details of this structure

EESEECRegistration ::= SEQUENCE

{

registrationType [1] RegistrationType,

eECID [2] UTF8String,

gPSI [3] GPSI OPTIONAL,

aCProfiles [4] ACProfiles OPTIONAL,

eECServiceContSupport [5] ACRScenarios OPTIONAL,

expirationTime [6] Timestamp OPTIONAL,

eECContextID [7] UTF8String OPTIONAL,

srcEESID [8] UTF8String OPTIONAL,

unfulfilledACProfiles [9] UnfulfilledACProfiles OPTIONAL,

failureResponse [10] FailureResponse OPTIONAL

}

-- See clause 7.14.2.3 for details of this structure

EESEASDiscovery ::= SEQUENCE

{

eECID [1] UTF8String,

gPSI [2] GPSI OPTIONAL,

eASDiscoveryFilter [3] EASDiscoveryFilter OPTIONAL,

eECServiceContSupport [4] ACRScenarios OPTIONAL,

uELocation [5] Location OPTIONAL,

eASTargetDNAIs [6] DNAIs OPTIONAL,

discoveredEAS [7] DiscoveredEAS OPTIONAL,

failureResponse [8] FailureResponse OPTIONAL

}

-- See clause 7.14.2.4 for details of this structure

EESEASDiscoverySubscription ::= SEQUENCE

{

eECID [1] UTF8String,

gPSI [2] GPSI OPTIONAL,

subscriptionType [3] SubscriptionType,

eASEventType [4] EASEventType,

eASDiscoveryFilter [5] EASDiscoveryFilter OPTIONAL,

eASDynamicInfoFilter [6] EASDynamicInfoFilter OPTIONAL,

eECServiceContSupport [7] ACRScenarios OPTIONAL,

expirationTime [8] Timestamp OPTIONAL,

subscriptionId [9] UTF8String OPTIONAL,

failureResponse [10] FailureResponse OPTIONAL

}

-- See clause 7.14.2.5 for details of this structure

EESEASDiscoveryNotification ::= SEQUENCE

{

subscriptionID [1] UTF8String,

eventType [2] EASEventType,

discoveredEAS [3] DiscoveredEAS,

failureResponse [4] FailureResponse OPTIONAL

}

-- See clause 7.14.2.6 for details of this structure

EESAppContextRelocation ::= SEQUENCE

{

eECID [1] UTF8String,

gPSI [2] GPSI OPTIONAL,

eESACRDetOrInit [3] EESACRDetOrInit

}

EESACRDetOrInit ::= CHOICE

{

aCRDetermineReq [1] ACRDetermineReq,

aCRInitiateReq [2] ACRInitiateReq

}

ACRDetermineReq ::= SEQUENCE

{

eASID [1] EASID OPTIONAL,

aCID [2] ACID OPTIONAL,

sEASEndpoint [3] EASEndpoint

}

ACRInitiateReq ::= SEQUENCE

{

eASID [1] EASID OPTIONAL,

aCID [2] ACID OPTIONAL,

tEASEndpoint [3] EASEndpoint,

sEASEndpoint [4] EASEndpoint OPTIONAL,

previousTEASEndpoint [5] EASEndpoint OPTIONAL,

routeReq [6] RouteToLocation OPTIONAL

}

-- See clause 7.14.2.7 for details of this structure

EESACRSubscription ::= SEQUENCE

{

eECID [1] UTF8String,

gPSI [2] GPSI OPTIONAL,

subscriptionType [3] SubscriptionType,

expirationTime [4] Timestamp OPTIONAL,

eASIDs [5] EASIDs,

aCIDs [6] ACIDs OPTIONAL,

eventIDs [7] ACREventIDs OPTIONAL,

subscriptionId [8] UTF8String OPTIONAL,

failureResponse [9] FailureResponse OPTIONAL

}

-- See clause 7.14.2.8 for details of this structure

EESACRNotification ::= SEQUENCE

{

subscriptionID [1] UTF8String,

eASID [2] EASID,

eventID [3] ACREventIDs,

targetInfo [4] TargetInfo OPTIONAL,

aCRRes [5] BOOLEAN OPTIONAL,

failReason [6] UTF8String OPTIONAL

}

-- See clause 7.14.2.9 for details of this structure

EESEECContextRelocation ::= SEQUENCE

{

eECID [1] UTF8String,

eECContextID [2] UTF8String,

gPSI [3] GPSI OPTIONAL,

uELoc [4] Location OPTIONAL,

aCProfiles [5] ACProfiles OPTIONAL

}

-- See clause 7.14.2.10 for details of this structure

EESStartOfInterceptionWithRegisteredEEC ::= SEQUENCE

{

eECID [1] UTF8String,

gPSI [2] GPSI OPTIONAL,

aCProfiles [3] ACProfiles OPTIONAL,

eECServiceContSupport [4] ACRScenarios OPTIONAL,

expirationTime [5] Timestamp OPTIONAL,

eECContextID [6] UTF8String OPTIONAL,

srcEESID [7] UTF8String OPTIONAL,

unfulfilledACProfiles [8] UnfulfilledACProfiles OPTIONAL,

timeOfRegistration [9] Timestamp OPTIONAL

}

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

-- EES parameters

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

RegistrationType ::= ENUMERATED

{

registration (1),

registrationUpdate(2),

deregistration(3)

}

ACProfiles ::= SET OF ACProfile

ACProfile ::= SEQUENCE

{

aCID [1] ACID,

aCType [2] UTF8String OPTIONAL,

aCSchedule [3] Daytime OPTIONAL,

expACGeoServArea [4] Location OPTIONAL,

eASsInfo [5] EASsInfo OPTIONAL,

aCServiceContSupport [6] ACRScenarios OPTIONAL

}

ACID ::= UTF8String

ACRScenarios ::= SET OF ACRScenario

ACRScenario ::= ENUMERATED

{

eECInitiated(1),

eECExecutedViaSourceEES(2),

eECExecutedViaTargetEES(3),

sourceEASDecided(4),

sourceEESExecuted(5),

eELManagedACR(6)

}

UnfulfilledACProfiles ::= SET OF UnfulfilledACProfile

UnfulfilledACProfile ::= SEQUENCE

{

aCID [1] ACID,

reason [2] UnfulfilledACProfileReason

}

UnfulfilledACProfileReason ::= ENUMERATED

{

eASNotAvailable(1),

requirementsUnfulfilled(2)

}

EASID ::= UTF8String

EASsInfo ::= SET OF EASInfo

EASInfo ::= SEQUENCE

{

eASID [1] EASID,

expectedSvcKPIs [2] ServiceKPIs OPTIONAL,

minimumReqSvcKPIs [3] ServiceKPIs OPTIONAL

}

ServiceKPIs ::= SEQUENCE

{

connectionBandwidth [1] INTEGER OPTIONAL,

requestRate [2] INTEGER OPTIONAL,

responseTime [3] INTEGER OPTIONAL,

requestedAvailability [4] INTEGER OPTIONAL,

requestedCompute [5] OCTET STRING OPTIONAL,

requestedGraphCompute [6] OCTET STRING OPTIONAL,

requestedMemory [7] OCTET STRING OPTIONAL,

requestedStorage [8] OCTET STRING OPTIONAL

}

FailureResponse ::= ENUMERATED

{

error400(1),

error401(2),

error403(3),

error404(4),

error406(5),

error411(6),

error413(7),

error415(8),

error429(9),

error500(10),

error503(11)

}

EASDiscoveryFilter ::= CHOICE

{

aCsCharacteristics [1] ACProfiles,

eASCharacteristics [2] EASsCharacteristics

}

EASsCharacteristics ::= SET OF EASCharacteristics

EASCharacteristics ::= SEQUENCE

{

eASID [1] EASID OPTIONAL,

aSPID [2] UTF8String OPTIONAL,

eASType [3] UTF8String OPTIONAL,

eASSchedule [4] Daytime OPTIONAL,

eASProfile [5] EASProfile OPTIONAL,

eASServiceArea [6] Location OPTIONAL,

eASServicePermLevel [7] UTF8String OPTIONAL,

eASServiceFeatures [8] EASServiceFeatures OPTIONAL

}

DNAIs ::= SET OF DNAI

DiscoveredEAS ::= SEQUENCE

{

eASProfile [1] EASProfile,

lifetime [2] INTEGER OPTIONAL

}

EASProfile ::= SEQUENCE

{

eASID [1] EASID,

eASEndpoint [2] EASEndpoint,

aCIDs [3] ACIDs OPTIONAL,

aSPID [4] UTF8String OPTIONAL,

eASType [5] UTF8String OPTIONAL,

eASDescription [6] UTF8String OPTIONAL,

eASSchedule [7] Daytime OPTIONAL,

eASServiceArea [8] Location OPTIONAL,

eASServiceKPIs [9] ServiceKPIs OPTIONAL,

eASServicePermLevel [10] UTF8String OPTIONAL,

eASServiceFeatures [11] EASServiceFeatures OPTIONAL,

eASServiceContSupport [12] ACRScenarios OPTIONAL,

appLocs [13] RouteToLocations OPTIONAL,

eASStatus [14] EASStatus OPTIONAL

}

EASStatus ::= ENUMERATED

{

enabled(1),

disabled(2)

}

EASEndpoint ::= SEQUENCE

{

fQDN [1] FQDN OPTIONAL,

iPv4Addresses [2] IPv4Addresses OPTIONAL,

iPv6Addresses [3] IPv6Addresses OPTIONAL,

uRI [4] UTF8String OPTIONAL

}

RouteToLocations ::= SET OF RouteToLocation

EASServiceFeatures ::= SET OF EASServiceFeature

EASServiceFeature ::= UTF8String

ACIDs ::= SET OF ACID

IPv4Addresses ::= SET OF IPv4Address

IPv6Addresses ::= SET OF IPv6Address

SubscriptionType ::= ENUMERATED

{

subscription(1),

subscriptionUpdate(2),

unsubscription(3)

}

EASEventType ::= ENUMERATED

{

eASAvailabilityChange(1),

eASDynamicInfoChange(2)

}

EASDynamicInfoFilter ::= SEQUENCE

{

eASId [1] EASID,

eASStatus [2] BOOLEAN,

eASAcIDs [3] BOOLEAN,

eASDesc [4] BOOLEAN,

eASPt [5] BOOLEAN,

eASFeature [6] BOOLEAN,

eASSchedule [7] BOOLEAN,

eASSvcArea [8] BOOLEAN,

eASSvcKpi [9] BOOLEAN,

eASSvcCont [10] BOOLEAN

}

EASIDs ::= SET OF EASID

ACREventIDs ::= ENUMERATED

{

targetInformation(1),

aCRComplete(2)

}

TargetInfo ::= SEQUENCE

{

discoveredEAS [1] DiscoveredEAS,

targetEESInfo [2] EDNConfigurationInfo OPTIONAL

}

EDNConfigurationInfo ::= SEQUENCE

{

eDNConnectionInfo [1] EDNConnectionInfo,

eESsInfo [2] EESsInfo,

lifetime [3] INTEGER OPTIONAL

}

EDNConnectionInfo ::= SEQUENCE

{

dNN [1] DNN OPTIONAL,

sNSSAI [2] SNSSAI OPTIONAL,

serviceArea [3] Location OPTIONAL

}

EESsInfo ::= SET OF EESInfo

EESInfo ::= SEQUENCE

{

eESID [1] EESID,

eESEndpoint [2] EESEndpoint,

eASIDs [3] EASIDs OPTIONAL,

serviceArea [4] Location OPTIONAL,

dNAIS [5] DNAIs OPTIONAL

}

EESID ::= UTF8String

EESEndpoint ::= SEQUENCE

{

fQDN [1] FQDN OPTIONAL,

iPv4Addresses [2] IPv4Addresses OPTIONAL,

iPv6Addresses [3] IPv6Addresses OPTIONAL,

uRI [4] UTF8String OPTIONAL

}

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

-- 5GMS AF definitions

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

-- See clause 7.15.2.2 for details of this structure

FiveGMSAFServiceAccessInformation ::= SEQUENCE

{

gPSI [1] GPSI,

serviceAccessInformationResource [2] SBIType

}

-- See clause 7.15.2.3 for details of this structure

FiveGMSAFConsumptionReporting ::= SEQUENCE

{

gPSI [1] GPSI,

consumptionReport [2] SBIType

}

-- See clause 7.15.2.4 for details of this structure

FiveGMSAFDynamicPolicyInvocation ::= SEQUENCE

{

gPSI [1] GPSI,

dynamicPolicyResource [2] SBIType,

dPIoperationType [3] DPIOperationType

}

-- See clause 7.15.2.5 for details of this structure

FiveGMSAFMetricsReporting ::= SEQUENCE

{

gPSI [1] GPSI,

metricsReport [2] XMLType

}

-- See clause 7.15.2.6 for details of this structure

FiveGMSAFNetworkAssistance ::= SEQUENCE

{

gPSI [1] GPSI,

networkAssistanceSessionResource [2] SBIType,

nAOperationType [3] NAOperationType

}

-- See clause 7.15.2.7 for details of this structure

FiveGMSAFUnsuccessfulProcedure ::= SEQUENCE

{

gPSI [1] GPSI,

fiveGMSAFUnsuccessfulOperation [2] FiveGMSAFUnsuccessfulOperation,

fiveGMSAFErrorCode [3] FiveGMSAFErrorCode

}

-- See clause 7.15.2.8 for details of this structure

FiveGMSAFStartOfInterceptionWithAlreadyConfiguredUE ::= SEQUENCE

{

gPSI [1] GPSI,

serviceAccessInformationResource [2] SBIType

}

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

-- 5GMS AF parameters

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

DPIOperationType ::= ENUMERATED

{

createDynamicPolicy(1),

retrieveDynamicPolicy(2),

updateDynamicPolicy(3),

patchDynamicPolicy(4),

destroyDynamicPolicy(5)

}

NAOperationType ::= ENUMERATED

{

createNetworkAssistanceSession(1),

retrieveNetworkAssistanceSession(2),

updateNetworkAssistanceSession(3),

patchNetworkAssistanceSession(4),

destroyNetworkAssistanceSession(5),

requestBitRateRecommendation(6),

requestDeliveryBoost(7)

}

FiveGMSAFUnsuccessfulOperation::= ENUMERATED

{

retrieveServiceAccessInformation(1),

submitConsumptionReport(2),

submitMetricsReport(3),

createDynamicPolicy(4),

retrieveDynamicPolicy(5),

updateDynamicPolicy(6),

patchDynamicPolicy(7),

destroyDynamicPolicy(8),

createNetworkAssistanceSession(9),

retrieveNetworkAssistanceSession(10),

updateNetworkAssistanceSession(11),

patchNetworkAssistanceSession(12),

destroyNetworkAssistanceSession(13),

requestBitRateRecommendation(14),

requestDeliveryBoost(15)

}

FiveGMSAFErrorCode ::=ENUMERATED

{

badRequest400(1),

unauthorized401(2),

notFound404(3),

unsupportedMediaType415(4)

}

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

-- 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,

useSessionTrigger [14] BOOLEAN

}

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

-- 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),

modifyEPSBearerContectReject(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

}

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

-- HSS definitions

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

HSSServingSystemMessage ::= SEQUENCE

{

iMSI [1] IMSI,

oldPLMNID [2] PLMNID,

newPLMNID [3] PLMNID,

roamingIndicator [4] RoamingIndicator,

responseCodes [5] UTF8String

}

HSSStartOfInterceptionWithRegisteredTarget ::= SEQUENCE

{

hSSIdentities [1] HSSIdentities,

subscriptionDataSets [2] SubscriptionDataSets,

pSUserState [3] SBIType

}

HSSIdentities ::= SEQUENCE

{

ePSSubscriberIDs [1] EPSSubscriberIDs OPTIONAL,

iMSSubscriberIDs [2] IMSSubscriberIDs OPTIONAL

}

SubscriptionDataSets ::= CHOICE

{

iMSSubscriptionData [1] SBIType

}

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

-- Common Parameters

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

AccessType ::= ENUMERATED

{

threeGPPAccess(1),

nonThreeGPPAccess(2),

threeGPPandNonThreeGPPAccess(3)

}

AllowedNSSAI ::= SEQUENCE OF NSSAI

AllowedTACs ::= SEQUENCE (SIZE(1..MAX)) OF TAC

AreaOfInterest ::= SEQUENCE

{

areaOfInterestTAIList [1] AreaOfInterestTAIList OPTIONAL,

areaOfInterestCellList [2] AreaOfInterestCellList OPTIONAL,

areaOfInterestRANNodeList [3] AreaOfInterestRANNodeList OPTIONAL

}

AreaOfInterestCellList ::= SEQUENCE (SIZE(1..MAX)) OF NCGI

AreaOfInterestItem ::= SEQUENCE

{

areaOfInterest [1] AreaOfInterest

}

AreaOfInterestRANNodeList ::= SEQUENCE (SIZE(1..MAX)) OF GlobalRANNodeID

AreaOfInterestTAIList ::= SEQUENCE (SIZE(1..MAX)) OF TAI

CellCAGList ::= SEQUENCE (SIZE(1..MAX)) OF CAGID

CauseMisc ::= ENUMERATED

{

controlProcessingOverload(1),

notEnoughUserPlaneProcessingResources(2),

hardwareFailure(3),

oMIntervention(4),

unknownPLMNOrSNPN(5),

unspecified(6)

}

CauseNas ::= ENUMERATED

{

normalRelease(1),

authenticationFailure(2),

deregister(3),

unspecified(4)

}

CauseProtocol ::= ENUMERATED

{

transferSyntaxError(1),

abstractSyntaxError-reject(2),

abstractSyntaxErrorIgnoreAndNotify(3),

messageNotCompatibleWithReceiverState(4),

semanticError(5),

abstractSyntaxErrorFalselyConstructedMessage(6),

unspecified(7)

}

CauseRadioNetwork ::= ENUMERATED

{

unspecified(1),

txnrelocoverallExpiry(2),

successfulHandover(3),

releaseDueToNGRANGeneratedReason(4),

releaseDueTo5gcGeneratedReason(5),

handoverCancelled(6),

partialHandover(7),

hoFailureInTarget5GCNGRANNodeOrTargetSystem(8),

hoTargetNotAllowed(9),

tNGRelocOverallExpiry(10),

tNGRelocPrepExpiry(11),

cellNotAvailable(12),

unknownTargetID(13),

noRadioResourcesAvailableInTargetCell(14),

unknownLocalUENGAPID(15),

inconsistentRemoteUENGAPID(16),

handoverDesirableForRadioReason(17),

timeCriticalHandover(18),

resourceOptimisationHandover(19),

reduceLoadInServingCell(20),

userInactivity(21),

radioConnectionWithUELost(22),

radioResourcesNotAvailable(23),

invalidQoSCombination(24),

failureInRadioInterfaceProcedure(25),

interactionWithOtherProcedure(26),

unknownPDUSessionID(27),

multiplePDUSessionIDInstances(29),

multipleQoSFlowIDInstances(30),

encryptionAndOrIntegrityProtectionAlgorithmsNotSupported(31),

nGIntraSystemHandoverTriggered(32),

nGInterSystemHandoverTriggered(33),

xNHandoverTriggered(34),

notSupported5QIValue(35),

uEContextTransfer(36),

iMSVoiceeEPSFallbackOrRATFallbackTriggered(37),

uPIntegrityProtectioNotPossible(38),

uPConfidentialityProtectionNotPossible(39),

sliceNotSupported(40),

uEInRRCInactiveStateNotReachable(41),

redirection(42),

resourcesNotAvailableForTheSlice(43),

uEMaxIntegrityProtectedDataRateReason(44),

releaseDueToCNDetectedMobility(45),

n26InterfaceNotAvailable(46),

releaseDueToPreemption(47),

multipleLocationReportingReferenceIDInstances(48),

rSNNotAvailableForTheUP(49),

nPMAccessDenied(50),

cAGOnlyAccessDenied(51),

insufficientUECapabilities(52)

}

CauseTransport ::= ENUMERATED

{

transportResourceUnavailable(1),

unspecified(2)

}

Direction ::= ENUMERATED

{

fromTarget(1),

toTarget(2)

}

DNN ::= UTF8String

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

EmailAddress ::= UTF8String

EquivalentPLMNs ::= SEQUENCE (SIZE(1..MAX)) OF PLMNID

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)

FiveGSSubscriberID ::= CHOICE

{

sUPI [1] SUPI,

sUCI [2] SUCI,

pEI [3] PEI,

gPSI [4] GPSI

}

FiveGSSubscriberIDs ::= SEQUENCE

{

fiveGSSubscriberID [1] SEQUENCE SIZE(1..MAX) OF FiveGSSubscriberID

}

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)

}

ForbiddenAreaInformation ::= SEQUENCE

{

pLMNIdentity [1] PLMNID,

forbiddenTACs [2] ForbiddenTACs

}

ForbiddenTACs ::= SEQUENCE (SIZE(1..MAX)) OF TAC

FTEID ::= SEQUENCE

{

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

iPv4Address [2] IPv4Address OPTIONAL,

iPv6Address [3] IPv6Address OPTIONAL

}

FTEIDList ::= SEQUENCE OF FTEID

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

}

HandoverCause ::= CHOICE

{

radioNetwork [1] CauseRadioNetwork,

transport [2] CauseTransport,

nas [3] CauseNas,

protocol [4] CauseProtocol,

misc [5] CauseMisc

}

HandoverType ::= ENUMERATED

{

intra5GS(1),

fiveGStoEPS(2),

ePSto5GS(3),

fiveGStoUTRA(4)

}

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

IMSSubscriberIDs ::= CHOICE

{

iMPI [1] IMPI,

iMPU [2] IMPU

}

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)

LocationAreaOfInterestList ::= SEQUENCE (SIZE(1..MAX)) OF AreaOfInterestItem

LocationEventType ::= ENUMERATED

{

direct(1),

changeOfServeCell(2),

uEPrescenceInAreaOfInterest(3),

stopChangeOfServeCell(4),

stopUEPresenceInAreaOfInterest(5),

cancelLocationReportingForTheUE(6)

}

LocationReportArea ::= ENUMERATED

{

cell(1)

}

LocationReportingRequestType ::= SEQUENCE

{

eventType [1] LocationEventType,

reportArea [2] LocationReportArea,

areaOfInterestList [3] LocationAreaOfInterestList

}

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

MobilityRestrictionList ::= SEQUENCE

{

servingPLMN [1] PLMNID,

equivalentPLMNs [2] EquivalentPLMNs OPTIONAL,

rATRestrictions [3] RATRestrictions OPTIONAL,

forbiddenAreaInformation [4] ForbiddenAreaInformation OPTIONAL,

serviceAreaInformation [5] ServiceAreaInformation OPTIONAL

}

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

NAI ::= UTF8String

NextLayerProtocol ::= INTEGER(0..255)

NonLocalID ::= ENUMERATED

{

local(1),

nonLocal(2)

}

NonIMEISVPEI ::= CHOICE

{

mACAddress [1] MACAddress,

eUI64 [2] EUI64

}

NPNAccessInformation ::= CHOICE

{

pNINPNAccessInformation [1] CellCAGList

}

NSSAI ::= SEQUENCE OF SNSSAI

PagingRestrictionIndicator ::= OCTET STRING (SIZE(1..33))

PLMNID ::= SEQUENCE

{

mCC [1] MCC,

mNC [2] MNC

}

PLMNList ::= SEQUENCE (SIZE(1..MAX)) OF PLMNID

PDNConnectionType ::= ENUMERATED

{

iPv4(1),

iPv6(2),

iPv4v6(3),

nonIP(4),

ethernet(5)

}

PDUSessionID ::= INTEGER (0..255)

PDUSessionResourceInformation ::= SEQUENCE

{

pDUSessionID [1] PDUSessionID

}

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)

RANUENGAPID ::= INTEGER (0..4294967295)

-- See clause 9.3.1.20 of TS 38.413 [23] for details

RANSourceToTargetContainer ::= OCTET STRING

-- See clause 9.3.1.21 of TS 38.413 [23] for details

RANTargetToSourceContainer ::= OCTET STRING

RATRestrictions ::= SEQUENCE (SIZE(1..MAX)) OF RATRestrictionItem

RATRestrictionInformation ::= BIT STRING (SIZE(8, ...))

RATRestrictionItem ::= SEQUENCE

{

pLMNIdentity [1] PLMNID,

rATRestrictionInformation [2] RATRestrictionInformation

}

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

ServiceAreaInformation ::= SEQUENCE (SIZE(1..MAX)) OF ServiceAreaInfo

ServiceAreaInfo ::= SEQUENCE

{

pLMNIdentity [1] PLMNID,

allowedTACs [2] AllowedTACs OPTIONAL,

notAllowedTACs [3] ForbiddenTACs OPTIONAL

}

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,

iMPU [11] IMPU,

iMPI [12] IMPI,

e164Number [13] E164Number,

emailAddress [14] EmailAddress,

mCPTTID [15] UTF8String,

instanceIdentifierURN [16] UTF8String,

pTCChatGroupID [17] PTCChatGroupID

}

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

}

UserIdentifiers ::= SEQUENCE

{

fiveGSSubscriberIDs [1] FiveGSSubscriberIDs OPTIONAL,

ePSSubscriberIDs [2] EPSSubscriberIDs OPTIONAL

}

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

-- 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..327675)

AgeOfLocationEstimate ::= INTEGER (0..32767)

HorizontalSpeed ::= UTF8String

VerticalSpeed ::= UTF8String

SpeedUncertainty ::= UTF8String

BarometricPressure ::= INTEGER (30000..115000)

-- 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)

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

-- Structures to allow reuse of externally defined structures

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

ReportedMessagePayload ::= CHOICE

{

mIMEEntityContainer [1] MIMEEntityContainer,

mSRPContainer [2] MSRPContainer

}

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

-- Externally Defined Structures - MSRP Parameters

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

EncapsulatedMSRP ::= OCTET STRING

MSRPContainer ::= CHOICE

{

encapsulatedMSRP [1] EncapsulatedMSRP,

separatedMSRP [2] SeparatedMSRP

}

MSRPEndLine ::= UTF8String

MSRPHeaders ::= UTF8String

MSRPStartLine ::= UTF8String

SeparatedMSRP ::= SEQUENCE

{

startLine [1] MSRPStartLine,

headers [2] MSRPHeaders,

body [3] MIMEEntityContainer OPTIONAL,

endLine [4] MSRPEndLine

}

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

-- Externally Defined Structures - SBIType Parameters

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

SBIContainer ::= SEQUENCE

{

includedSBIMessage [1] SBIType

}

SBIReference ::= UTF8String

-- Details for the encoding and use of this parameter may be found in the clause

-- that defines the xIRI that carries it. This parameter provides a generic

-- mechanism to convey service based interface structures defined in Stage 3 working groups.

SBIType ::= SEQUENCE

{

sBIReference [1] SBIReference,

sBIValue [2] SBIValue

}

SBIValue ::= UTF8String

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

-- Externally Defined Structures - XMLType Parameters

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

XMLContainer ::= SEQUENCE

{

includedXML [1] XMLType

}

XMLType ::= SEQUENCE

{

xMLNamespace [1] XMLNamespace,

xMLValue [2] XMLValue

}

XMLNamespace ::= UTF8String

XMLValue ::= UTF8String

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

-- Externally Defined Structures - MIME Entity Parameters

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

EncapsulatedMIMEEntity ::= OCTET STRING

MIMEBody ::= UTF8String

MIMEBodyLength ::= INTEGER

MIMEEntityBody ::= CHOICE

{

mIMEEntityContainer [1] MIMEEntityContainer,

bodyWithCommunicationsContent [2] MIMEEntityBodyWithCC,

multipartBody [3] MultiPartMIMEBody,

mSRPMessage [4] MSRPContainer,

xMLBody [5] XMLType,

sBIMessage [6] SBIType,

body [7] MIMEBody

}

MIMEEntityBodyWithCC ::= CHOICE

{

mIMEBodyWithOnlyContent [1] MIMEBodyLength,

mIMEBodyWithXML [2] XMLContainer,

mIMEBoodyWithSBIMessage [3] SBIContainer

}

MIMEContentType ::= UTF8String

MIMEEntityContainer ::= SEQUENCE

{

contentType [1] MIMEContentType,

mIMEEntity [2] MIMEEntity

}

MIMEEntity ::= CHOICE

{

encapsulatedMIMEEntity [1] EncapsulatedMIMEEntity,

separatedMIMEHeadersAndBody [2] SeparatedMIMEHeadersAndBody

}

MIMEHeaderSection ::= SEQUENCE

{

includedHeaderFields [1] MIMEHeaderFields,

removedHeaderFields [2] SEQUENCE OF RemovedMIMEHeaderField OPTIONAL

}

MIMEHeaderBodyLength ::= INTEGER

MIMEHeaderFieldName ::= UTF8String

MIMEHeaderHash ::= OCTET STRING

MIMEHeaderFields ::= UTF8String

MultiPartMIMEBody ::= SEQUENCE OF MIMEBodyPart

MIMEBodyPart ::= CHOICE

{

encapsulatedMIMEEntity [1] EncapsulatedMIMEEntity,

separatedMIMEBodyPart [2] SeparatedMIMEBodyPart

}

RemovedMIMEHeaderField ::= SEQUENCE

{

headerFieldName [1] MIMEHeaderFieldName,

headerBodyLength [2] MIMEHeaderBodyLength,

headerHash [3] MIMEHeaderHash OPTIONAL

}

SeparatedMIMEHeadersAndBody ::= SEQUENCE

{

entityHeaderSection [1] MIMEHeaderSection,

entityBody [2] MIMEEntityBody OPTIONAL

}

SeparatedMIMEBodyPart ::= SEQUENCE

{

bodyPartHeaderSection [1] MIMEHeaderSection OPTIONAL,

bodyPartContents [2] MIMEEntityBody OPTIONAL

}

END

## \*\*\*\* END OF ATTACHMENT CHANGES \*\*\*

## \*\*\*\* END OF ALL CHANGES \*\*\*