**3GPP TSG- Meeting #**

**, , -**

|  |
| --- |
| *CR-Form-v12.3* |
| **CHANGE REQUEST** |
|  |
|  |  | **CR** |  | **rev** |  | **Current version:** |  |  |
|  |
| *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 | **X** | Core Network | **X** |

|  |
| --- |
|  |
| ***Title:***  |  |
|  |  |
| ***Source to WG:*** |  |
| ***Source to TSG:*** | S5 |
|  |  |
| ***Work item code:*** |  |  | ***Date:*** |  |
|  |  |  |  |  |
| ***Category:*** |  |  | ***Release:*** |  |
|  | *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 canbe 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-17 (Release 17)Rel-18 (Release 18)Rel-19 (Release 19) Rel-20 (Release 20)* |
|  |  |
| ***Reason for change:*** | YANG SS is not matching the approved stage 2. |
|  |  |
| ***Summary of change:*** | Update YANG code to match existing stage 2. |
|  |  |
| ***Consequences if not approved:*** | Stage 2 and Stage 3 mismatch; interoperability problems. |
|  |  |
| ***Clauses affected:*** | Forge only |
|  |  |
|  | **Y** | **N** |  |  |
| ***Other specs*** |  | **X** |  Other core specifications  | TS/TR ... CR ...  |
| ***affected:*** |  | **X** |  Test specifications | TS/TR ... CR ...  |
| ***(show related CRs)*** |  | **X** |  O&M Specifications | TS/TR ... CR ...  |
|  |  |
| ***Other comments:*** | Forge MR link: <https://forge.3gpp.org/rep/sa5/MnS/-/merge_requests/1339> at commit 3f911d913dbd07e919d5901c46027b651a0ff6b4 |
|  |  |
| ***This CR's revision history:*** |  |

\*\*\* START OF CHANGE 2 \*\*\*

\*\*\* yang-models/\_3gpp-common-trace.yang \*\*\*

<CODE BEGINS>

module \_3gpp-common-trace {

 yang-version 1.1;

 namespace "urn:3gpp:sa5:\_3gpp-common-trace";

 prefix "trace3gpp";

 import \_3gpp-common-top { prefix top3gpp; }

 import \_3gpp-common-yang-types {prefix types3gpp; }

 import \_3gpp-common-yang-extensions {prefix yext3gpp; }

 import ietf-inet-types { prefix inet; }

 import \_3gpp-common-files { prefix files3gpp; }

 organization "3GPP SA5";

 contact "https://www.3gpp.org/DynaReport/TSG-WG--S5--officials.htm?Itemid=464";

 description "Trace handling

 Copyright 2024, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI,

 TTA, TTC). All rights reserved.";

 reference "3GPP TS 28.623

 Generic Network Resource Model (NRM)

 Integration Reference Point (IRP);

 Solution Set (SS) definitions

 3GPP TS 28.622

 Generic Network Resource Model (NRM)

 Integration Reference Point (IRP);

 Information Service (IS)" ;

 revision 2024-08-13 { reference CR-0387; }

 revision 2024-05-12 { reference "CR-0401"; }

 revision 2024-05-06 { reference CR-0359; }

 revision 2024-04-06 { reference "CR-0342"; }

 revision 2024-01-29 { reference "CR-0316"; }

 revision 2023-11-06 { reference "CR-0290 CR-0294"; }

 revision 2023-11-03 { reference CR-0302 ; }

 revision 2023-09-18 { reference CR-0271 ; }

 revision 2023-08-10 { reference CR-0261; }

 revision 2023-04-26 { reference CR-0250; }

 revision 2023-02-18 { reference "CR-0234"; }

 revision 2023-02-16 { reference "CR-0233"; }

 revision 2022-09-30 { reference CR-0191 ; }

 revision 2022-04-27 { reference "CR-0159"; }

 revision 2021-10-18 { reference "CR-0139"; }

 revision 2021-07-22 { reference "CR-0137"; }

 revision 2021-01-25 { reference "CR-0122"; }

 revision 2020-11-16 { reference "CR-0117"; }

 revision 2020-08-06 { reference "CR-0102"; }

 feature FilesUnderTraceJob {

 description "Files shall be contained under TraceJob";

 }

 grouping FreqInfoGrp {

 description "Represents the FreqInfo dataType.

 This <<dataType>> defines the RF reference frequency and the frequency

 operating bands used in a cell for a given direction (UL or DL) in FDD

 or for both UL and DL directions in TDD";

 leaf arfcn {

 type uint32 {

 range 0..3279165;

 }

 mandatory true;

 description "RF Reference Frequency as defined in TS 38.104,

 clause 5.4.2.1. The frequency provided identifies the absolute

 frequency position of the reference resource block (Common RB 0)

 of the carrier. Its lowest subcarrier is also known as Point A.";

 }

 leaf-list freqBands {

 type uint32 {

 range 1..1024;

 }

 min-elements 1;

 description "List of NR frequency operating bands. Primary NR

 Operating Band as defined in TS 38.104, clause 5.4.2.3.

 The value 1 corresponds to n1, value 2 corresponds to NR operating

 band n2, etc.";

 }

 }

 grouping AreaConfigGrp {

 description "Represents the AreaConfig dataType.

 This <<dataType>> defines the area for which measurement logging should

 be performed. It is described by a list of cells and a list of

 frequencies.";

 list freqInfo {

 key arfcn;

 min-elements 1;

 max-elements 32;

 description "It specifies the carrier frequency and bands used in

 a cell.";

 uses FreqInfoGrp ;

 }

 leaf-list pciList {

 type uint32 {

 range 0..1007;

 }

 min-elements 1;

 max-elements 32;

 description "List of neighbour cells subject for MDT scope.";

 }

 }

 grouping AreaScopeGrp {

 description "Represents the AreaScope dataType.

 This <<dataType>> defines the area scope of MDT.

 The Area Scope parameter in LTE and NR is either:

 - list of Cells, identified by E-UTRAN-CGI or NG-RAN CGI.

 Maximum 32 CGI can be defined.

 - list of Tracking Area, identified by TAC.

 Maximum of 8 TAC can be defined.

 - list of Tracking Area Identity, identified by TAC with

 associated plmn-Identity perTAC-List containing the

 PLMN identity for each TAC. Maximum of 8 TAI can be defined.

 The Area Scope parameter in NR can also contain:

 - list of NPN-IDs in NR. It is either a list of PNI-NPNs

 identified by CAG ID with associated plmn-Identity or a

 list of SNPNs identified by Network ID with associated

 plmn-Identity .";

 choice AreaScopeChoice {

 leaf-list eutraCellIdList {

 type string;

 min-elements 1;

 max-elements 32;

 description "List of E-UTRAN cells identified by E-UTRAN-CGI";

 }

 leaf-list utraCellIdList {

 type string;

 min-elements 1;

 max-elements 32;

 description "List of UTRAN cells identified by UTRAN CGI";

 }

 leaf-list tacList {

 type types3gpp:Tac;

 min-elements 1;

 max-elements 8;

 description "Tracking Area Code list";

 }

 list taiList {

 description "Tracking Area Identity list";

 key idx;

 min-elements 1;

 max-elements 8;

 leaf idx { type string; }

 uses types3gpp:TaiGrp;

 }

 }

 list nPNIdentityList {

 description "list of NPN IDs of in NR. It is either a list of PNI-NPNs

 identified by CAG ID with associated plmn-Identity or a list of SNPN

 identified by Network ID with associated plmn-Identity";

 key idx;

 min-elements 1;

 uses NpnIdGrp;

 leaf idx { type string; }

 }

 }

 grouping NpnIdGrp {

 description "Represents the NpnId dataType.";

 list plmnId {

 key "mcc mnc";

 description "It specifies the PLMN Id of the NPN network.";

 uses types3gpp:PLMNId;

 max-elements 1;

 }

 list cAGIdList {

 key idx;

 max-elements 256;

 description "It specifies the PNI-NPN identified by CAG ID ";

 leaf idx { type string; }

 }

 list nIDList {

 key idx;

 max-elements 16;

 description "It specifies the SNPN identified by Network ID";

 leaf idx { type string; }

 }

 }

 grouping ExcessPacketDelayThresholdsGrp {

 description "Represents the ExcessPacketDelayThresholds dataType.

 This <<dataType>> defines a excess packet delay threshold information

 to enable the calculation of the PDCP Excess Packet Delay in the

 uplink in case of M6 uplink measurements are requested. The excess

 packet delay threshold information is specified with the 5QI value

 and excess packet delay threshold value.";

 leaf fiveQIValue {

 type uint8;

 mandatory true;

 description "It indicates 5QI value.";

 }

 leaf excessPacketDelayThresholdValue {

 type decimal64 {

 fraction-digits 2;

 range 0.25|0.5|1|2|4|5|10|20|30|40|50|60|70|80|90|100|150|300|500 ;

 }

 mandatory true;

 units milliseconds;

 description "Value of excess packet delay threshold

 for M6 UL measurement in milliseconds.";

 }

 }

 grouping TraceReferenceGrp {

 description "Represents the TraceReference dataType.

 This <<dataType>> defines a globally unique identifier, which uniquely

 identifies the Trace Session that is created by the TraceJob. It is

 composed of the MCC, MNC (resulting in PLMN identifier) and the

 trace identifier.";

 uses types3gpp:PLMNId; // mcc+mnc

 leaf traceId {

 type string;

 mandatory true;

 description "An identifier, which identifies the Trace

 (together with MCC and MNC). This is a 3 byte Octet String.";

 }

 }

 grouping MbsfnAreaGrp {

 description "Represents the MbsfnArea dataType.

 This <<dataType>> defines a MBSFN area. It is composed of the MBSFN Area

 identifier and the carrier frequency (EARFCN).";

 leaf mbsfnAreaId {

 type uint32 {

 range 1..max;

 }

 mandatory true;

 description "MBSFN Area Identifier";

 }

 leaf earfcn{

 type uint32 {

 range 1..max;

 }

 mandatory true;

 description "Carrier Frequency";

 }

 }

 grouping TraceConfigGrp {

 description "Defines the configuration parameters of TraceJob

 which are specific for Trace or combined Trace and Immediate MDT.

 The attribute listOfNeTypes specifies the network elements to be

 traced. The optional attribute listOfInterfaces allows to specify

 the individual interfaces of the network elements to be recorded.

 The attribute traceDepth allows to configure the level of detail

 of the information which shall be recorded. For trace the reporting

 is event based, where the triggering event is configured with

 attribute triggeringEvent. For each triggering event the first and

 last message (start/stop triggering event) to record are specified.";

 list listOfInterfaces {

 key idx;

 description "Specifies the interfaces that need to be traced in the given

 ManagedEntityFunction.The attribute is applicable only for Trace. In

 case this attribute is not used, it carries a null semantic.";

 reference "Clause 5.5 of 3GPP TS 32.422 for additional details on the

 allowed values.";

 leaf idx { type uint32 ; }

 leaf-list MSCServerInterfaces {

 type enumeration {

 enum A ;

 enum Iu-CS ;

 enum Mc ;

 enum MAP-G ;

 enum MAP-B ;

 enum MAP-E ;

 enum MAP-F ;

 enum MAP-D ;

 enum MAP-C ;

 enum CAP ;

 }

 }

 leaf-list MGWInterfaces {

 type enumeration {

 enum Mc ;

 enum Nb-UP ;

 enum Iu-UP ;

 }

 }

 leaf-list RNCInterfaces {

 type enumeration {

 enum Iu-CS ;

 enum Iu-PS ;

 enum Iur ;

 enum Iub ;

 enum Uu ;

 }

 }

 leaf-list SGSNInterfaces {

 type enumeration {

 enum Gb ;

 enum Iu-PS ;

 enum Gn ;

 enum MAP-Gr ;

 enum MAP-Gd ;

 enum MAP-Gf ;

 enum Ge ;

 enum Gs ;

 enum S6d ;

 enum S4 ;

 enum S3 ;

 enum S13 ;

 }

 }

 leaf-list GGSNInterfaces {

 type enumeration {

 enum Gn ;

 enum Gi ;

 enum Gmb ;

 }

 }

 leaf-list S-CSCFInterfaces {

 type enumeration {

 enum Mw ;

 enum Mg ;

 enum Mr ;

 enum Mi ;

 }

 }

 leaf-list P-CSCFInterfaces {

 type enumeration {

 enum Gm ;

 enum Mw ;

 }

 }

 leaf-list I-CSCFInterfaces {

 type enumeration {

 enum Cx ;

 enum Dx ;

 enum Mg ;

 enum Mw ;

 }

 }

 leaf-list MRFCInterfaces {

 type enumeration {

 enum Mp ;

 enum Mr ;

 }

 }

 leaf-list MGCFInterfaces {

 type enumeration {

 enum Mg ;

 enum Mj ;

 enum Mn ;

 }

 }

 leaf-list IBCFInterfaces {

 type enumeration {

 enum Ix ;

 enum Mx ;

 }

 }

 leaf-list E-CSCFInterfaces {

 type enumeration {

 enum Mw ;

 enum Ml ;

 enum Mm ;

 enum Mi-Mg ;

 }

 }

 leaf-list BGCFInterfaces {

 type enumeration {

 enum Mi ;

 enum Mj ;

 enum Mk ;

 }

 }

 leaf-list ASInterfaces {

 type enumeration {

 enum Dh ;

 enum Sh ;

 enum ISC ;

 enum Ut ;

 }

 }

 leaf-list HSSInterfaces {

 type enumeration {

 enum MAP-C ;

 enum MAP-D ;

 enum Gc ;

 enum Gr ;

 enum Cx ;

 enum S6d ;

 enum S6a ;

 enum Sh ;

 }

 }

 leaf-list EIRInterfaces {

 type enumeration {

 enum MAP-F ;

 enum S13 ;

 enum MAP-Gf ;

 }

 }

 leaf-list BM-SCInterfaces {

 type enumeration {

 enum Gmb ;

 }

 }

 leaf-list MMEInterfaces {

 type enumeration {

 enum S1-MME ;

 enum S3 ;

 enum S6a ;

 enum S10 ;

 enum S11 ;

 enum S13 ;

 }

 }

 leaf-list SGWInterfaces {

 type enumeration {

 enum S4 ;

 enum S5 ;

 enum S8 ;

 enum S11 ;

 enum Gxc ;

 }

 }

 leaf-list PDN\_GWInterfaces {

 type enumeration {

 enum S2a ;

 enum S2b ;

 enum S2c ;

 enum S5 ;

 enum S6b ;

 enum Gx ;

 enum S8 ;

 enum SGi ;

 }

 }

 leaf-list eNBInterfaces {

 type enumeration {

 enum S1-MME ;

 enum X2 ;

 }

 }

 leaf-list en-gNBInterfaces {

 type enumeration {

 enum S1-MME ;

 enum X2 ;

 enum Uu ;

 enum F1-C ;

 enum E1 ;

 }

 }

 leaf-list AMFInterfaces {

 type enumeration {

 enum N1 ;

 enum N2 ;

 enum N8 ;

 enum N11 ;

 enum N12 ;

 enum N14 ;

 enum N15 ;

 enum N20 ;

 enum N22 ;

 enum N26 ;

 }

 }

 leaf-list AUSFInterfaces {

 type enumeration {

 enum N12 ;

 enum N13 ;

 }

 }

 leaf-list NEFInterfaces {

 type enumeration {

 enum N29 ;

 enum N30 ;

 enum N33 ;

 }

 }

 leaf-list NRFInterfaces {

 type enumeration {

 enum N27 ;

 }

 }

 leaf-list NSSFInterfaces {

 type enumeration {

 enum N22 ;

 enum N31 ;

 }

 }

 leaf-list PCFInterfaces {

 type enumeration {

 enum N5 ;

 enum N7 ;

 enum N15 ;

 }

 }

 leaf-list SMFInterfaces {

 type enumeration {

 enum N4 ;

 enum N7 ;

 enum N10 ;

 enum N11 ;

 enum S5-C ;

 enum N38 ;

 enum N16 ;

 enum N16a ;

 }

 }

 leaf-list SMSFInterfaces {

 type enumeration {

 enum N20 ;

 enum N21 ;

 }

 }

 leaf-list UDMInterfaces {

 type enumeration {

 enum N8 ;

 enum N10 ;

 enum N13 ;

 enum N21 ;

 }

 }

 leaf-list UPFInterfaces {

 type enumeration {

 enum N4 ;

 }

 }

 leaf-list ng-eNBInterfaces {

 type enumeration {

 enum NG-C ;

 enum Xn-C ;

 enum Uu ;

 }

 }

 leaf-list gNB-CU-CPInterfaces {

 type enumeration {

 enum NG-C ;

 enum Xn-C ;

 enum Uu ;

 enum F1-C ;

 enum E1 ;

 enum X2-C ;

 }

 }

 leaf-list gNB-CU-UPInterfaces {

 type enumeration {

 enum E1 ;

 }

 }

 leaf-list gNB-DUInterfaces {

 type enumeration {

 enum F1-C ;

 }

 }

 }

 leaf-list listOfNETypes {

 type enumeration {

 enum MSC\_SERVER;

 enum SGSN;

 enum MGW;

 enum GGSN;

 enum RNC;

 enum BM\_SC;

 enum MME;

 enum SGW;

 enum PGW;

 enum ENB;

 enum EN\_GNB;

 enum GNB\_CU\_CP;

 enum GNB\_CU\_UP;

 enum GNB\_DU;

 }

 description "Specifies in which type of ManagedFunction the trace should

 be activated. The attribute is applicable only for Trace with

 Signalling Based Trace activation. In case this attribute is not used,

 it carries a null semantic";

 reference "Clause 5.4 of 3GPP TS 32.422 for additional details on the

 allowed values";

 }

 leaf traceDepth {

 when '../../jobType = "TRACE\_ONLY"'

 + ' or ../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type enumeration {

 enum MINIMUM;

 enum MEDIUM;

 enum MAXIMUM;

 enum VENDORMINIMUM;

 enum VENDORMEDIUM;

 enum VENDORMAXIMUM;

 }

 default MAXIMUM;

 description "Specifies how detailed information should be recorded in the

 Network Element. The Trace Depth is a paremeter for Trace Session level,

 i.e., the Trace Depth is the same for all of the NEs to be traced in

 the same Trace Session.

 The attribute is applicable only for Trace, otherwise it carries a null

 semantic.";

 reference "Clause 5.3 of 3GPP TS 32.422";

 }

 leaf triggeringEvents {

 type enumeration {

 enum MO\_MT\_CALLS;

 enum MO\_MT\_SMS;

 enum LU\_IMSIattach\_IMSIdetach;

 enum HANDOVER;

 enum SS;

 enum PDPcontext;

 enum RAU\_GPRSattach\_GPRSdetach;

 enum MBMScontext;

 enum CONTEXT;

 enum SIPsession\_StandaloneTransaction;

 enum MBMSactivation;

 enum UEinitiatedPDNconnectivityRequest;

 enum ServiceRequest;

 enum InitialAttach\_TAU\_Detach;

 enum UEinitiatedPDNdisconnection;

 enum BearerActivationModificationDeletion;

 enum Handover;

 enum PDNconnectionCreation;

 enum PDNconnectionTermination;

 enum Registration;

 enum UEderegistration;

 enum NetworkDeregistration;

 enum UEMobilityFromEPC;

 enum UEMobilityToEPC;

 enum PDUsessionEstablishment;

 enum PDUsessionModification;

 enum PDUsessionRelease;

 enum PDUsessionUPactivationDeactivation;

 enum MobilityBtw3gppAndN3gppTo5GC;

 enum MobilityFromEpc;

 enum AMpolicy;

 enum SMpolicy;

 enum Authorization;

 enum BDTpolicy;

 enum N4Session;

 enum UEauthentication;

 enum EventExposure;

 enum PFDmanagement;

 enum ParameterProvision;

 enum Trigger;

 enum NFmanagement;

 enum NFdiscovery;

 enum NSSelection;

 enum NSSAI;

 enum SMservice;

 enum UEcontext;

 enum SubscriberData;

 }

 description "It specifies the triggering event parameter of the trace

 session. The attribute is applicable only for Trace. In case this

 attribute is not used, it carries a null semantic.

 See the clause 5.1 of 3GPP TS 32.422 for additional details on the

 allowed values.";

 }

 }

 grouping ImmediateMdtConfigGrp {

 description "Represents the ImmediateMdtConfig dataType.

 This <<dataType>> defines the configuration parameters of

 IOC TraceJob which are specific for Immediate MDT or combine

 Trace and Immediate MDT.

 The optional attribute positioningMethod allows to specify

 the positioning methods to use.

 The following attributes are conditional available based on the

 measurements configured in listOfMeasurements:

 -reportInterval: conditional for M1 in LTE or NR and M1/M2 in UMTS,

 -reportAmount: conditional for M1 in LTE or NR and M1/M2 in UMTS,

 -reportingTrigger: conditional for M1 in LTE or NR and M1/M2 in UMTS,

 -eventThreshold: conditional for A2 event reporting or A2 event

 triggered periodic reporting,

 -collectionPeriodRRMNR: conditional for M4 and M5 in NR,

 -collectionPeriodM6NR: conditional for M6 in NR,

 -collectionPeriodM7NR: conditional for M7 in NR,

 -collectionPeriodRRMLTE (conditional for M3 in LTE),

 -measurementPeriodLTE (conditional for M4 and M5 in LTE),

 -collectionPeriodM6LTE (conditional for M6 in LTE),

 -collectionPeriodM7LTE (conditional for M7 in LTE),

 -collectionPeriodRRMUMTS (conditional for M4 and M5 in UMTS),

 -measurementPeriodUMTS (conditional for M6 and M7 in UMTS),

 -measurementQuantity (conditional for 1F event reporting),

 -beamLevelMeasurement (conditional for M1 in NR),

 -excessPacketDelayThresholds (conditional for M6 UL measurement in NR).

 For immediate MDT, the measurement reporting is dependent on the

 configured measurements:

 - For measurement M1 in LTE or NR, it is possible to select between

 periodical, event triggered, event triggered periodic reporting or

 reporting according to all configured RRM event triggers. For M1 and M2

 measurement in UMTS, it is possible to select between periodical, event

 triggered reporting or reporting according to all configured RRM event

 triggers. Parameter reportingTrigger determines which of the reporting

 methods is selected and in case of event triggered or event-triggered

 periodic, which is the decisive event type. For periodical reporting,

 parameters reportInterval and reportAmount determine the interval between

 two successive reports and the number of reports. This means the

 periodical reporting terminates after reportAmount reports have been

 sent as long as reportAmount is configured with a value different from

 infinity. For event-triggered periodic reporting, these two parameters

 apply in addition to parameter eventThreshold which determines the

 threshold of the event. In this case up to reportAmount reports are

 sent with a periodicity of reportInterval after the entering condition

 is fulfilled. The reporting is stopped, if the leaving condition is

 fulfulled and is restarted if the configured event reoccurs. For event

 based reporting, there is only one report sent after the event occurs.

 The parameters to configure are reportingTrigger and eventThreshold.

 In case of UMTS and 1f event reporting, additionally parameter

 measurementQuantity is necessary in order to determine for which

 measurement(s) the event threshold is applicable. Parameter

 beamLevelMeasurement determines whether beam level measurements shall

 be included in case of NR.

 - For measurement M2 in LTE or NR, reporting is according to RRM

 configuration, see TS 38.321, TS 36.321 and TS 38.331, TS 36.331.

 For measurement M4 in UMTS, reporting is either according to RRM

 configuration, see TS 25.321 and TS 25.331 or periodic or event

 triggered periodic using parameter collectionPeriodRRMUMTS and

 eventThresholdUphUMTS.

 - For measurement M3 in UMTS, the reporting is done upon

 availability, see TS 37.320.

 - For measurements M4, M5, M6 and M7 in NR, for measurements

 M3, M4, M5, M6 and M7 in LTE and for measurements M5, M6 and M7

 in UMTS periodical reporting is applied. The configurable parameter

 is the interval between two measurements (collectionPeriodRRMNR,

 collectionPeriodM6NR, collectionPeriodM7NR, collectionPeriodRRMLTE,

 measurementPeriodLte, collectionPeriodM6LTE, collectionPeriodM7LTE,

 collectionPeriodRRMUMTS, measurementPeriodUMTS). If no collection

 period is configured for M5 in UMTS, all available measurements are

 logged according to RRM configuration.";

 leaf listOfMeasurements {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"';

 type enumeration {

 enum M1;

 enum M2;

 enum M3;

 enum M4;

 enum M5;

 enum M6\_DL;

 enum M6\_UL;

 enum M7\_DL;

 enum M7\_UL;

 enum M1\_EVENT\_TRIGGERED;

 enum M6;

 enum M7;

 enum M8;

 enum M9;

 }

 description "It specifies the UE measurements that shall be collected in

 an Immediate MDT job. The attribute is applicable only for Immediate MDT.

 In case this attribute is not used, it carries a null semantic.";

 reference "3GPP TS 32.422 clause 5.10.3";

 }

 leaf reportingTrigger {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"';

 type enumeration {

 enum PERIODICAL;

 enum A2\_FOR\_LTE;

 enum 1F\_FOR\_UMTS;

 enum 1I\_FOR\_UMTS\_MCPS\_TDD;

 enum A2\_TRIGGERED\_PERIODIC\_FOR\_LTE;

 enum ALL\_CONFIGURED\_RRM\_FOR\_LTE;

 enum ALL\_CONFIGURED\_RRM\_FOR\_UMTS;

 }

 description "It specifies whether periodic or event based measurements

 should be collected.

 The attribute is applicable only for Immediate MDT and when the

 listOfMeasurements is configured for M1 (for both UMTS and LTE)

 or M2 (only for UMTS). In case this attribute is not used, it carries

 a null semantic.";

 reference "Clause 5.10.4 of 3GPP TS 32.422";

 }

 leaf reportInterval {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' and ../reportingTrigger = "PERIODICAL"';

 type uint32 {

 range "120|240|250|480|500|640|1000|1024|2000|2048|3000|4000|"

 +"5120|6000|8000|10240|12000|16000|20000|"

 +"20480|24000|28000|32000|40960|60000|64000|"

 +"360000|720000|1800000|3600000";

 }

 units milliseconds;

 description "It specifies the interval between the periodical measurements

 that shall be taken when the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and when

 reportingTrigger is configured for periodical measurements. In case

 this attribute is not used, it carries a null semantic.";

 reference "5.10.5 of 3GPP TS 32.422";

 }

 leaf reportAmount {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' and ../reportingTrigger = "PERIODICAL"';

 type union {

 type uint32 {

 range "1|4|8|16|32|64" ;

 }

 type enumeration {

 enum INFINITY;

 }

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected.

 The attribute is applicable only for Immediate MDT and when

 reportingTrigger is configured for periodical measurements. In

 case this attribute is not used, it carries a null semantic.";

 reference "Clause 5.10.6 of 3GPP TS 32.422";

 }

 leaf eventThreshold {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"';

 type int64;

 description "Specifies the threshold which should trigger the reporting

 in case A2 event reporting in LTE or 1F/1l event in UMTS. The attribute

 is applicable only for Immediate MDT and when reportingTrigger is

 configured for A2 event in LTE or 1F event or 1l event in UMTS. In

 case this attribute is not used, it carries a null semantic.";

 reference "Clauses 5.10.7 and 5.10.7a of 3GPP TS 32.422";

 }

 leaf collectionPeriodRRMNR {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint32 {

 range "1024|2048|5120|10240|60000";

 }

 units milliseconds;

 description "Specifies the collection period for collecting RRM

 configured measurement samples for M4, M5 in NR. The attribute is

 applicable only for Immediate MDT. In case this attribute is not

 used, it carries a null semantic.";

 reference "Clause 5.10.30 of 3GPP TS 32.422";

 }

 leaf collectionPeriodM6NR {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type enumeration {

 enum 120ms;

 enum 240ms;

 enum 480ms;

 enum 640ms;

 enum 1024ms;

 enum 2048ms;

 enum 5120ms;

 enum 10240ms;

 enum 20480ms;

 enum 40960ms;

 enum 1min;

 enum 6min;

 enum 12min;

 enum 30min;

 }

 description "It specifies the collection period for the Packet Delay

 measurement (M6) for NR MDT taken by the gNB. The attribute is

 applicable only for Immediate MDT. In case this attribute is not used,

 it carries a null semantic.";

 reference "clause 5.10.34 of TS 32.422";

 }

 leaf collectionPeriodM7NR {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint32 {

 range "1..60";

 }

 description "It specifies the collection period for the Packet Loss Rate

 measurement (M7) for NR MDT taken by the gNB. The attribute is

 applicable only for Immediate MDT. In case this attribute is not used,

 it carries a null semantic.";

 reference "clause 5.10.35 of TS 32.422";

 }

 leaf collectionPeriodRRMLTE {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint32 {

 range "250|500|1000|2000|3000|4000|6000|8000|12000|16000|20000|"

 +"24000|28000|32000|64000";

 }

 units milliseconds;

 description "Specifies the collection period for collecting RRM configured

 measurement samples for M2, M3 in LTE. The attribute is applicable only

 for Immediate MDT. In case this attribute is not used, it carries a

 null semantic.";

 reference "Clause 5.10.20 of 3GPP TS 32.422";

 }

 leaf measurementPeriodLTE {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint32 {

 range "1024|1280|2048|2560|5120|"

 +"10240|60000";

 }

 units milliseconds;

 mandatory true;

 description "It specifies the measurement period for the Data Volume and

 Scheduled IP throughput measurements for MDT taken by the eNB.

 The attribute is applicable only for Immediate MDT. In case this

 attribute is not used, it carries a null semantic.";

 reference "Clause 5.10.23 of 3GPP TS 32.422";

 }

 leaf collectionPeriodM6LTE {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint32 {

 range "1024|2048|5120|10240";

 }

 units milliseconds;

 description "Specifies the collection period for the Packet Delay

 measurement (M6) for MDT taken by the eNB. The attribute is applicable

 only for Immediate MDT. In case this attribute is not used,

 it carries a null semantic.";

 reference "Clause 5.10.32 of TS 32.422 ";

 }

 leaf collectionPeriodM7LTE {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint16 {

 range 1..60 ;

 }

 description "It specifies the collection period for the Packet Loss Rate

 measurement (M7) for LTE MDT taken by the eNB. The attribute is

 applicable only for Immediate MDT. In case this attribute

 is not used, it carries a null semantic.";

 reference "Clause 5.10.33 of TS 32.422 .";

 }

 leaf eventThresholdUphUMTS {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint16 {

 range 0..31 ;

 }

 description "It specifies the threshold which should trigger

 the reporting in case of event-triggered periodic reporting for M4

 (UE power headroom measurement) in UMTS. In case this attribute is

 not used, it carries a null semantic.";

 reference "5.10.39 of TS 32.422";

 }

 leaf collectionPeriodRRMUMTS {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint32 {

 range "1024|1280|2048|2560|5120|"

 +"10240|60000";

 }

 units milliseconds;

 description "Specifies the collection period for collecting RRM configured

 measurement samples for M3, M4, M5 in UMTS. The attribute is applicable

 only for Immediate MDT. In case this attribute is not used, it carries

 a null semantic";

 reference "Clause 5.10.21 of 3GPP TS 32.422";

 }

 leaf measurementPeriodUMTS {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + 'or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type uint32 {

 range "250|500|1000|2000|3000|4000|6000|8000|12000|16000|20000|"

 +"24000|28000|32000|64000";

 }

 units milliseconds;

 mandatory true;

 description "It specifies the measurement period for the Data Volume and

 Throughput measurements for MDT taken by RNC.

 The attribute is applicable only for Immediate MDT. In case this

 attribute is not used, it carries a null semantic.";

 reference "Clause 5.10.22 of 3GPP TS 32.422";

 }

 leaf measurementQuantity {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type enumeration {

 enum CPICH\_ECNO;

 enum CPICH\_RSCP;

 enum PATHLOSS;

 }

 description "It specifies the measurements that are collected in an MDT

 job for a UMTS MDT configured for event triggered reporting.";

 reference "Clause 5.10.15 of 3GPP TS 32.422";

 }

 leaf beamLevelMeasurement {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type boolean;

 default false;

 description "Indicates whether the NR M1 beam level measurements shall

 be included or not.";

 reference "Clause 5.10.40 of TS 32.422";

 }

 leaf positioningMethod {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../../../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type enumeration {

 enum GNSS;

 enum E\_CELL\_ID;

 }

 description "It specifies what positioning method should be used in the

 MDT job.";

 reference "Clause 5.10.19 of 3GPP TS 32.422";

 }

 list excessPacketDelayThresholds {

 description "Excess packet delay thresholds info for M6 UL measurement.";

 min-elements 1;

 key idx;

 leaf idx { type string; }

 uses ExcessPacketDelayThresholdsGrp;

 }

 leaf reportAmountM1LTE {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for LTE. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM4LTE {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for LTE. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM5LTE {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for LTE. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM6LTE {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for LTE. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM7LTE {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for LTE. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM1NR {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for NR. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM4NR {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for NR. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM5NR {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for NR. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM6NR {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for NR. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 leaf reportAmountM7NR {

 type enumeration {

 enum 1;

 enum 2;

 enum 4;

 enum 8;

 enum 16;

 enum 32;

 enum 64;

 enum INFINITY;

 }

 description "It specifies the number of measurement reports that shall be

 taken for periodic reporting while the UE is in connected mode.

 The attribute is applicable only for Immediate MDT and combined Trace

 and Immediate MDT and when reportingTrigger is configured for periodical

 measurements and applicable only for NR. In case this attribute is not

 used, it carries a null semantic.

 See the clause 5.10.6 of TS 32.422 for additional details on the

 allowed values.";

 }

 }

 grouping LoggedMdtConfigGrp {

 description "This <<dataType>> defines the configuration parameters of

 IOC TraceJob which are specific for Logged MDT or Logged MBSFN MDT.

 The optional attribute plmnList allows to specify the PLMNs where

 measurement collection, status indication and log reporting is allowed,

 the optional attribute areaConfigurationForNeighCell allows to specify

 the area for which UE is requested to perform measurements logging for

 neighbour cells which have list of frequencies. For logged MDT in UMTS

 and LTE, the reporting is periodical. Parameter loggingInterval determines

 the interval between the reports and parameter loggingDuration determines

 how long the configuration is valid meaning after this duration has passed

 no further reports are sent. In NR, the reporting can be periodical or event

 based, determined by parameter reportType. For periodical reporting the

 same parameters as in LTE and UMTS apply. For event based reporting,

 parameter eventListForEventTriggeredMeasurement configures the event type,

 namely 'out of coverage' or 'L1 event'. In case 'L1 event' is selected as

 event type, the logging is performed according to parameter loggingInterval

 at regular intervals only when the conditions indicated by eventThresholdL1,

 hysteresisL1, timeToTriggerL1 (defining the thresholds, hysteresis and time

 to trigger) are met and if UE is 'camped normally' state (TS 38.331,

 TS 38.304). In case 'out of coverage' is selected as event type, the

 logging is performed according to parameter loggingInterval at regular

 intervals only when the UE is in 'any cell selection' state.

 Furthermore, logging is performed immediately upon transition from the

 'any cell selection' state to the 'camped normally' state (TS 38.331,

 TS 38.304).";

 leaf traceCollectionEntityId {

 type int64;

 description "It specifies the TCE Id which is sent to the UE in

 Logged MDT. See the clause 5.10.11 of 3GPP TS 32.422 for additional

 details on the allowed values.";

 }

 leaf loggingDuration {

 when '../../../jobType = "LOGGED\_MDT\_ONLY" or'

 + ' ../../../jobType = "LOGGED\_MBSFN\_MDT"';

 type uint32 {

 range "600|1200|2400|3600|5400|7200";

 }

 units seconds;

 description "Specifies how long the MDT configuration is valid at the

 UE in case of Logged MDT. The attribute is applicable only for

 Logged MDT and Logged MBSFN MDT. In case this attribute is not used, it

 carries a null semantic.";

 reference "5.10.9 of 3GPP TS 32.422";

 }

 leaf loggingInterval {

 when '../../../jobType = "LOGGED\_MDT\_ONLY" or'

 + ' ../../../jobType = "LOGGED\_MBSFN\_MDT"';

 type uint32 {

 range "0|320|640|1280|2560|5120|10240|20480|"

 +"30720|40960|61440";

 }

 units milliseconds;

 description "Specifies the periodicty for Logged MDT. The attribute is

 applicable only for Logged MDT and Logged MBSFN MDT. In case this

 attribute is not used, it carries a null semantic.

 The value 0 indicates Infinity for NR.";

 reference "5.10.8 of 3GPP TS 32.422";

 }

 leaf reportType {

 when '../../../jobType = "IMMEDIATE\_MDT\_ONLY"';

 type enumeration {

 enum PERIODICAL;

 enum EVENT\_TRIGGERED;

 }

 description "It specifies report type for logged NR MDT";

 reference "Clause 5.10.27 of 3GPP TS 32.422";

 }

 leaf eventListForEventTriggeredMeasurement {

 when '../../../jobType = "LOGGED\_MDT\_ONLY"';

 type enumeration {

 enum OUT\_OF\_COVERAGE ;

 enum A2\_EVENT ;

 }

 description "Specifies event types for event triggered measurement in the

 case of logged NR MDT. Each trace session may configure at most one

 event. The UE shall perform logging of measurements only upon certain

 condition being fulfilled:

 - Out of coverage.

 - A2 event.";

 reference "Clause 5.10.28 of 3GPP TS 32.422";

 }

 leaf eventThresholdL1 {

 when '../../../jobType = "LOGGED\_MDT\_ONLY" or'

 + ' ../../../jobType = "LOGGED\_MBSFN\_MDT"';

 type uint32 {

 range "0..127";

 }

 description "It specifies the threshold which should trigger

 the reporting in case of event based reporting of logged NR MDT.

 The attribute is applicable only for Logged MDT and when reportType

 is configured for event triggered reporting and when

 eventListForEventTriggeredMeasurement is configured for L1 event.

 In case this attribute is not used, it carries a null semantic.";

 reference "clause 5.10.36 of TS 32.422";

 }

 leaf hysteresisL1 {

 when '../../../jobType = "LOGGED\_MDT\_ONLY" or '

 + '../../../jobType = "LOGGED\_MBSFN\_MDT"';

 type uint32 {

 range "0..30";

 }

 description "It specifies the hysteresis used within the entry and leave

 condition of the L1 event based reporting of logged NR MDT.

 The attribute is applicable only for Logged MDT, when reportType

 is configured for event triggered reporting and when

 eventListForEventTriggeredMeasurement is configured for L1 event.

 In case this attribute is not used, it carries a null semantic.";

 reference "clause 5.10.37 of TS 32.422";

 }

 leaf timeToTriggerL1 {

 when '../../../jobType = "LOGGED\_MDT\_ONLY" or '

 + '../../../jobType = "LOGGED\_MBSFN\_MDT"';

 type int32 {

 range 0|40|64|80|100|128|160|256|320|480|512|640|1024|1280|2560|5120;

 }

 units milliseconds;

 description "It specifies the threshold which should trigger

 the reporting in case of event based reporting of logged NR MDT.

 The attribute is applicable only for Logged MDT, when reportType

 is configured for event triggered reporting and when

 eventListForEventTriggeredMeasurement is configured for L1 event.

 In case this attribute is not used, it carries a null semantic.";

 reference "clauses 5.10.38 of TS 32.422";

 }

 list pLMNList {

 when '../../../jobType = "LOGGED\_MDT\_ONLY"';

 key "mcc mnc";

 uses types3gpp:PLMNId;

 max-elements 16;

 description "It indicates the PLMNs where measurement collection, status

 indication and log reporting is allowed.";

 reference "Clause 5.10.24 of 3GPP TS 32.422";

 }

 list areaConfigurationForNeighCells {

 when '../../../jobType = "LOGGED\_MDT\_ONLY"';

 key "idx";

 leaf idx { type uint32 ; }

 description "It specifies the area for which UE is requested to perform

 measurement logging for neighbour cells which have list of frequencies.

 If it is not configured, the UE shall perform measurement logging for

 all the neighbour cells.

 Applicable only to NR Logged MDT.";

 reference "3GPP TS 32.422 clause 5.10.26.";

 uses AreaConfigGrp;

 }

 list mbsfnAreaList {

 when '../../../jobType = "LOGGED\_MBSFN\_MDT"';

 key "mbsfnAreaId earfcn";

 max-elements 8;

 description "The MBSFN Area consists of a MBSFN Area ID and Carrier

 Frequency (EARFCN). The target MBSFN area List can have up to 8 entries.

 This parameter is applicable only if the job type is Logged MBSFN MDT.";

 reference "5.10.25 of 3GPP TS 32.422";

 uses MbsfnAreaGrp;

 }

 list nPNIdentityList {

 description "It defines which NPNs that can be served by the NR cell,

 and which CAG IDs or NIDs can be supported by the NR cell for

 corresponding PNI-NPN or SNPN.";

 key idx;

 max-elements 1;

 uses NpnIdGrp;

 leaf idx { type string;}

 }

 }

 grouping MdtConfigGrp {

 description "Defines the configuration parameters of IOC

 TraceJob which are specific for MDT. The attribute

 anonymizationOfMdtData specifies the level of anonymization

 of MDT data. The optional attribute areaScope allows to

 specify the area in terms of cells or Tracking Area/Routing

 Area/Location area where the MDT data collection shall take

 place. In case of RLF\_REPORT\_ONLY and RCEF\_REPORT\_ONLY the

 optional attribute areaScope allows to specify the eNB or list

 of eNBs or gNB or list of gNBs where the reports should be

 collected. The optional attribute sensorInformation allows to

 specify the sensor information to include. Based on the value

 configured for attribute jobType in IOC TraceJob, the attributes

 immediateMdtConfig or loggedMdtConfig are available: In case of

 IMMEDIATE\_MDT\_ONLY or IMMEDIATE\_MDT\_AND\_TRACE the attribute

 immediateMdtConfig is applicable. In case of LOGGED\_MDT\_ONLY or

 LOGGED\_MBSFN\_MDT the attribute loggedMdtConfig is applicable.";

 leaf anonymizationOfMDTData {

 when ../areaScope ;

 type enumeration {

 enum NO\_IDENTITY;

 enum TAC\_OF\_IMEI;

 }

 default NO\_IDENTITY;

 description "Specifies level of MDT anonymization.";

 reference "3GPP TS 32.422 clause 5.10.12.";

 }

 list areaScope {

 key "idx";

 leaf idx { type uint32 ; }

 description "It specifies the area where data shall be collected.

 List of eNB/list of gNB/eNB/gNB for RLF or RCEF.

 List of cells/TA/LA/RA for signaling based MDT or management

 based Logged MDT.

 List of cells for management based Immediate MDT.

 List of NPN IDs for management based MDT.

 Cell, TA, LA, RA are mutually exclusive.

 This attribute shall be present if MDT is supported.";

 reference "Clause 5.10.2 of 3GPP TS 32.422";

 uses AreaScopeGrp;

 }

 leaf-list sensorInformation {

 type enumeration {

 enum BAROMETRIC\_PRESSURE;

 enum UE\_SPEED;

 enum UE\_ORIENTATION;

 }

 description "It specifies which sensor information shall be included in

 logged NR MDT and immediate NR MDT measurement if they are available.

 The following sensor measurement can be included or excluded for

 the UE.";

 reference "Clause 5.10.29 of 3GPP TS 32.422";

 }

 list immediateMdtConfig {

 description "The set of parameters specific for Immediate MDT

 configuration.

 This attribute shall be present only if MDT is supported and the jobType

 attribute is set to Immediate MDT or combined Trace and Immediate MDT";

 key idx;

 max-elements 1;

 leaf idx { type string; }

 uses ImmediateMdtConfigGrp;

 }

 list loggedMdtConfig {

 description "The set of parameters specific for Logged MDT and Logged

 MBSFN MDT configuration.

 This attribute shall be present only if MDT is supported and the

 jobType attribute is set to Logged MDT or Logged MBSFN MDT.";

 key idx;

 max-elements 1;

 leaf idx { type string; }

 uses LoggedMdtConfigGrp;

 }

 }

 grouping UECoreMeasConfigGrp {

 description "Represents the ueCoreMeasConfig dataType.

 This <<dataType>> defines the aconfiguration parameters of IOC TraceJob

 which are specific for UE level measurements collection.";

 leaf-list ueCoreMeasurements {

 type string;

 min-elements 1;

 description "List of 5GC UE level measurements identified by name.

 The list may include 5GC UE level measurements defined in TS 28.558,

 or vendor specific measurements.

 For 5GC UE level measurements defined in TS 28.558, the name is

 constructed as follows:

 - 'family.measurementName.subcounter' for measurement type with

 specified subcounter

 - 'family.measurementName.ALL' for measurement type with all supported

 subcounters

 - 'family.measurementName' for measurement type without subcounters

 - 'family' for measurement family, including all measurement types and

 the associated subcounters under this family.

 For non-3GPP specified 5GC UE level measurements the name is defined

 elsewhere.";

 }

 leaf ueCoreMeasGranularityPeriod {

 type uint32;

 mandatory true;

 units milliseconds;

 description "It specifies the Granularity period used to produce UE level

 measurements. The period is defined in milliseconds (ms).";

 }

 leaf nfTypeToMeasure {

 type string;

 mandatory true;

 description "It indicates the type of NE to produce the 5GC UE level

 measurements.

 allowedValues: The NF types represented by the measured object

 classes as defined by f) of the 5GC UE level measurements specified

 in TS 28.558.";

 }

 leaf-list objectInstances {

 type string;

 description "List of object instances.";

 }

 leaf-list rootObjectInstances {

 type string;

 description "List of root object instances.";

 }

 }

 grouping TraceJobGrp {

 leaf jobType {

 type enumeration {

 enum IMMEDIATE\_MDT\_ONLY;

 enum LOGGED\_MDT\_ONLY;

 enum TRACE\_ONLY;

 enum IMMEDIATE\_MDT\_AND\_TRACE;

 enum RLF\_REPORT\_ONLY;

 enum RCEF\_REPORT\_ONLY;

 enum LOGGED\_MBSFN\_MDT;

 enum 5GC\_UE\_LEVEL\_MEASUREMENTS\_ONLY;

 enum TRACE\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS;

 enum IMMEDIATE\_MDT\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS;

 enum TRACE\_AND\_IMMEDIATE\_MDT\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS;

 }

 default TRACE\_ONLY;

 description "Specifies the MDT mode and it specifies also whether the

 TraceJob represents only MDT, Logged MBSFN MDT, Trace, or 5GC UE

 level measurement collection, or any combination

 of Trace, immediate MDT and 5GC UE level measurement collection.

 The attribute is applicable for Trace, MDT, RCEF and

 RLF reporting, and 5GC UE level measurement collection.";

 reference "Clause 5.9a of 3GPP TS 32.422 for additional details on the

 allowed values.";

 }

 list pLMNTarget {

 key "mcc mnc";

 description "Specifies which PLMN that the subscriber of the session to

 be recorded uses as selected PLMN. PLMN Target might differ from the

 PLMN specified in the Trace Reference";

 reference "Clause 5.9b of 3GPP TS 32.422";

 uses types3gpp:PLMNId;

 }

 leaf-list listOfTraceMetrics {

 when '../jobType = "TRACE\_ONLY"'

 + ' or ../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"';

 type string;

 description "Specifies the metrics to be reported.";

 reference "Clause 10 of 3GPP TS 32.422";

 }

 leaf traceCollectionEntityIPAddress {

 type inet:ip-address;

 description "It specifies the address of the Trace Collection Entity

 when the attribute traceReportingFormat is configured for the

 file-based reporting. The attribute is applicable for both Trace and

 MDT.

 See the clause 5.9 of TS 32.422 for additional details on the allowed

 values.";

 }

 leaf traceReportingConsumerUri {

 when '../traceReportingFormat = "STREAMING"';

 type inet:uri;

 description "URI of the Streaming Trace data reporting MnS consumer

 (a.k.a. streaming target).";

 reference "Clause 5.9 of 3GPP TS 32.422";

 }

 list traceReference {

 key "idx";

 min-elements 1;

 max-elements 1;

 description "A globally unique identifier, which uniquely identifies the

 Trace Session that is created by the TraceJob.

 In case of shared network, it is the MCC and MNC of the Participating

 Operator that request the trace session that shall be provided.

 The attribute is applicable for both Trace and MDT.";

 reference "Clause 5.6 of 3GPP TS 32.422";

 leaf idx { type uint32 ; }

 uses trace3gpp:TraceReferenceGrp ;

 }

 leaf jobId {

 type string;

 yext3gpp:inVariant;

 description "Identifier of a TraceJob";

 yext3gpp:inVariant;

 }

 leaf traceReportingFormat {

 type enumeration {

 enum FILE\_BASED;

 enum STREAMING;

 }

 default FILE\_BASED;

 description "Specifies the trace reporting format - streaming trace

 reporting or file-based trace reporting";

 reference "3GPP TS 32.422 clause 5.11";

 }

 list traceTarget {

 key "targetIdType targetIdValue";

 max-elements 1;

 leaf targetIdType {

 type enumeration {

 enum IMSI;

 enum IMEI;

 enum IMEISV;

 enum PUBLIC\_ID;

 enum UTRAN\_CELL;

 enum E\_UTRAN\_CELL;

 enum NG\_RAN\_CELL;

 enum ENB;

 enum RNC;

 enum GNB;

 enum SUPI;

 }

 }

 leaf targetIdValue {

 type string;

 }

 description "Specifies the target object of the Trace and MDT. The

 attribute is applicable for both Trace and MDT. This attribute

 includes the ID type of the target as an enumeration and the ID value.

 The traceTarget shall be public ID in case of a Management Based

 Activation is done to an ScscfFunction. The traceTarget shall be

 cell only in case of the UTRAN cell traffic trace function.

 The traceTarget shall be E-UtranCell only in case of E-UTRAN cell

 traffic trace function.The traceTarget shall be either IMSI or

 IMEI(SV) if the Trace Session is activated to any of the following

 ManagedEntity(ies):

 - HssFunction

 - MscServerFunction

 - SgsnFunction

 - GgsnFunction

 - BmscFunction

 - RncFunction

 - MmeFunction

 The traceTarget shall be IMSI if the Trace Session is activated to a

 ManagedEntity playing a role of ServinGWFunction.

 In case of signaling based Trace/MDT, the traceTarget attribute shall

 be able to carry (IMSI or IMEI(SV)or SUPI), the mDTAreaScope attribute

 shall be able to carry a list of (cell or E-UtranCell or NRCellDU or

 TA/LA/RA).

 In case of management based Immediate MDT, the traceTarget attribute

 shall be null value, the mDTAreaScope attribute shall carry a list of

 (Utrancell or E-UtranCell or NRCellDU).

 In case of management based Logged MDT, the traceTarget attribute

 shall carry an eBs or a RNC or gNBs. The Logged MDT should be initiated

 on the specified eNB or RNC or gNB in traceTarget. The mDTAreaScope

 attribute shall carry a list of (Utrancell or E-UtranCell or NRCellDU or

 TA/LA/RA).

 In case of RLF reporting, or RCEF reporting, the traceTarget

 attribute shall be null value, the mDTAreaScope attribute shall carry

 one or list of eNBs/gNBs";

 reference "3GPP TS 32.422";

 }

 list traceConfig {

 when '../jobType = "TRACE\_ONLY"'

 + ' or ../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"'

 + ' or ../jobType = "TRACE\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS"'

 + ' or ../jobType =

 "TRACE\_AND\_IMMEDIATE\_MDT\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS"';

 key idx;

 description "Trace config";

 max-elements 1;

 uses TraceConfigGrp;

 leaf idx { type string; }

 }

 list mdtConfig {

 when '../jobType = "IMMEDIATE\_MDT\_ONLY"'

 + ' or ../jobType = "IMMEDIATE\_MDT\_AND\_TRACE"'

 + ' or ../jobType = "RLF\_REPORT\_ONLY"'

 + ' or ../jobType = "RCEF\_REPORT\_ONLY"'

 + ' or ../jobType = "LOGGED\_MBSFN\_MDT"'

 + ' or ../jobType = "IMMEDIATE\_MDT\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS"'

 + ' or ../jobType =

 "TRACE\_AND\_IMMEDIATE\_MDT\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS"';

 key idx;

 description "MDT config";

 max-elements 1;

 uses MdtConfigGrp;

 leaf idx { type string; }

 }

 list ueCoreMeasConfig {

 when '../jobType = "5GC\_UE\_LEVEL\_MEASUREMENTS\_ONLY"'

 + ' or ../jobType = "TRACE\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS"'

 + ' or ../jobType = "IMMEDIATE\_MDT\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS"'

 + ' or ../jobType =

 "TRACE\_AND\_IMMEDIATE\_MDT\_AND\_5GC\_UE\_LEVEL\_MEASUREMENTS"';

 key idx;

 description "5GC UE level measurements config";

 max-elements 1;

 uses UECoreMeasConfigGrp;

 leaf idx { type string; }

 }

 list nPNTarget {

 description "applicable only for NR and shall be present in case of NPN

 either a PNI-NPN or a SNPN) and for management-based activation when

 several NPNs are supported in the RAN.";

 key idx;

 max-elements 1;

 uses NpnIdGrp;

 leaf idx { type string;}

 }

 }

 grouping TraceSubtree {

 description "Contains classes that manage Tracing.

 Should be used in all classes (or classes inheriting from)

 - SubNnetwork

 - ManagedElement

 - ManagedFunction

 If a YANG module wants to augment these classes/list/groupings they must

 augment all user classes!";

 list TraceJob {

 description "A TraceJob instance represents the Trace Control and

 Configuration parameters of a particular Trace Job (see TS 32.421 and

 TS 32.422 for details). It can be name-contained by SubNetwork,

 ManagedElement, ManagedFunction.

 To activate Trace Jobs, a MnS consumer has to create TraceJob object

 instances on the MnS producer. A MnS consumer can activate a Trace Job

 for another MnS consumer since it is not required the value of

 traceCollectionEntityIPAddress or traceReportingConsumerUri to be

 his own.

 For the details of Trace Job activation see clauses 4.1.1.1.2 and

 4.1.2.1.2 of TS 32.422.

 When a MnS consumer wishes to deactivate a Trace Job, the MnS consumer

 shall delete the corresponding TraceJob instance.

 For details of management Trace Job activation/deactivation see clause

 4.1.1.1.2 of TS 32.422.

 The attribute traceReference specifies a globally unique ID and

 identifies a Trace session. One Trace Session may be activated to

 multiple Network Elements. The traceReference is populated by the

 consumer that makes the request for a Trace Session.

 The jobId attribute presents the job identifier of a TraceJob instance.

 The jobId can be used to associate multiple TraceJob instances.

 For example, it is possible to configure the same jobId value for

 multiple TraceJob instances required to produce the data (e.g. RSRP

 values of M1 and RLF reports) for a specific network analysis.

 The attribute traceReportingFormat defines the method for reporting

 the produced measurements. The selectable options are file-based or

 stream-based reporting. In case of file-based reporting the attribute

 traceCollectionEntityIPAddress is used to specify the IP address to

 which the trace records shall be transferred, while in case of

 stream-based reporting the attribute traceReportingConsumerUri

 specifies the streaming target.

 The mandatory attribute traceTarget determines the target object of

 the TraceJob. Dependent on the network element to which the Trace

 Session is activated different types of the target object are possible.

 The attribute pLMNTarget defines the PLMN for which sessions shall be

 selected in the Trace Session in case of management based activation

 when several PLMNs are supported in the RAN.

 The attribute jobType specifies the kind of data to collect. In case of

 Trace only, the configuration parameters of attribute traceConfig shall

 be applied. In case of Immediate MDT only, Logged MDT only, RLF reports

 only, RCEF reports only and Logged MBSFN MDT, the configuration

 parameters of attribute mdtConfig or a subset of these shall be

 applied. In case of UE measurements only, the configuration parameters

 of attribute ueCoreMeasConfig shall be applied. In case of any combination

 of Trace, Immediate MDT, Trace and UE measurements, the configuration

 parameters of the corresponding attributes traceConfig, mdtConfig and

 ueCoreMeasConfig are applicable.

 Creation and deletion of TraceJob instances by MnS consumers is

 optional; when not supported, the TraceJob instances may be created

 and deleted by the system or be pre-installed.";

 key id;

 uses top3gpp:Top\_Grp ;

 container attributes {

 uses TraceJobGrp ;

 }

 uses files3gpp:FilesSubtree {

 if-feature FilesUnderTraceJob;

 }

 }

 }

}

<CODE ENDS>

\*\*\* END OF CHANGE 2 \*\*\*

\*\*\* START OF CHANGE 3 \*\*\*

\*\*\* yang-models/\_3gpp-common-util.yang \*\*\*

<CODE BEGINS>

module \_3gpp-common-util {

 yang-version 1.1;

 namespace urn:3gpp:sa5:\_3gpp-common-util;

 prefix "util3gpp";

 import \_3gpp-common-top { prefix top3gpp; }

 import \_3gpp-common-yang-types { prefix types3gpp ; }

 import \_3gpp-common-managed-element { prefix me3gpp ; }

 import \_3gpp-common-subnetwork { prefix subnet3gpp ; }

 organization "3GPP SA5";

 contact "https://www.3gpp.org/DynaReport/TSG-WG--S5--officials.htm?Itemid=464";

 description "Defines Utility IOCs: ConditionMonitor, Scheduler

 Copyright 2024, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI,

 TTA, TTC). All rights reserved.";

 reference "3GPP TS 28.623";

 revision 2024-08-13 { reference CR-0387; }

 revision 2024-05-06 { reference CR-0359; }

 feature SchedulerUnderManagedElement {

 description "Scheduler shall be contained under ManagedElement.";

 }

 feature SchedulerUnderSubNetwork {

 description "Scheduler shall be contained under SubNetwork.";

 }

 grouping TimeIntervalGrp {

 description "This data type defines a time interval within one day.

 If the whole day shall be selected, intervalStart shall be set to

 00:00:00 and intervalEnd shall be set to 23:59:59.";

 leaf intervalStart {

 type types3gpp:FullTime;

 mandatory true;

 description "It indicates the time (in 'full-time' format) when the

 service shall be started.

 Data type 'FullTime' defines the time as specified by 'full-time'

 in RFC3339.";

 }

 leaf intervalEnd {

 type types3gpp:FullTime;

 mandatory true;

 description "It indicates the time (in 'full-time' format) when the

 service shall be stopped.

 'FullTime' defines the time as specified by 'full-time' in RFC3339.

 intervalEnd should use the same timezone as intervalStart and shall

 be later then intervalStart";

 }

 }

 grouping SchedulingTimeGrp {

 description "This <<choice>> defines the scheduling time and allows to

 configure one of four possible scheduling methods:

 - One time interval: The attribute timeWindow presents the active

 scheduling time. A duration more than one day may be configured.

 - Daily periodicity: Several active intervals per day can be configured

 in attribute timeIntervals. The active scheduling times recur each day.

 - Weekly periodicity: Several active intervals for one day can be

 configured in attribute timeIntervals. The active scheduling times recur

 on the days of the weeks configured by attribute daysOfWeek

 - Monthly periodicity: Several active intervals for one day can be

 configured in attribute timeIntervals. The active scheduling times

 recur on the days of the months configured by attribute daysOfMonth.";

 choice SchedulingTime {

 mandatory true;

 case One-time-interval {

 uses types3gpp:TimeWindowGrp;

 }

 case periodic {

 list timeIntervals {

 description "List of intervals within one day for which the service

 shall be active.";

 key "intervalStart intervalEnd";

 min-elements 1;

 uses TimeIntervalGrp;

 }

 leaf daysOfWeek {

 must 'not(../daysOfMonth)';

 description "Shall be present in case of Weekly periodicity,

 otherwise not";

 type types3gpp:DayOfWeek;

 }

 leaf daysOfMonth {

 must 'not(../daysOfWeek)';

 description "Shall be present in case of Monthly periodicity,

 otherwise not";

 type types3gpp:DateMonthDay;

 }

 }

 }

 }

 grouping SchedulerGrp {

 description "Represents the Scheduler IOC.";

 list schedulingTimes {

 description "It defines the active scheduling times.";

 min-elements 1;

 key idx;

 leaf idx { type uint32; }

 uses SchedulingTimeGrp;

 }

 leaf schedulerStatus {

 type boolean;

 mandatory true;

 config false;

 description "Switches between TRUE and FALSE depending upon whether the

 configured time constraints are fulfilled or not.";

 }

 }

 grouping SchedulerSubtree {

 list Scheduler {

 description "This IOC defines a time scheduler.

 It can be name-contained by SubNetwork or ManagedElement.

 The attribute schedulingTimes allows to configure one or several

 active time intervals. The active intervals can be configured to occur

 once or recurring periodically.

 The boolean attribute schedulerStatus switches between TRUE and FALSE

 depending upon whether the configured time constraints are fulfilled

 or not. This attribute makes the internal Scheduler status observable.";

 key id;

 uses top3gpp:Top\_Grp;

 container attributes {

 uses SchedulerGrp;

 }

 }

 }

 augment /me3gpp:ManagedElement {

 if-feature SchedulerUnderManagedElement;

 uses SchedulerSubtree;

 }

 augment /subnet3gpp:SubNetwork {

 if-feature SchedulerUnderSubNetwork;

 uses SchedulerSubtree;

 }

///////////////////////////////////////////////////////////////////////////////

 feature ConditionMonitorUnderManagedElement {

 description "ConditionMonitor shall be contained under ManagedElement.";

 }

 feature ConditionMonitorUnderSubNetwork {

 description "ConditionMonitor shall be contained under SubNetwork.";

 }

 grouping ConditionMonitorGrp {

 description "Represents the Scheduler IOC.";

 leaf condition {

 type string {

 length 1..max;

 }

 mandatory true;

 description "Logical expression of one or several condition(s).

 A restricted XPath may be used. The definition of the exact syntax

 is not ready yet.

 An empty string is not allowed.";

 }

 leaf conditionStatus {

 type boolean;

 mandatory true;

 config false;

 description "Switches between TRUE and FALSE depending upon whether the

 configured constraints are fulfilled or not.";

 }

 }

 grouping ConditionMonitorSubtree {

 list ConditionMonitor {

 description "This IOC defines one or several conditions and monitors

 whether these conditions are satisfied. It can be name-contained by

 SubNetwork or ManagedElement.

 The attribute condition allows to configure one or several conditions.

 Possible conditions include but are not limited to scheduling

 requirements or parameter settings e.g. evaluation if a configuration

 parameter is above a certain threshold or has a certain values.

 The boolean attribute conditionStatus switches between TRUE and FALSE

 depending upon whether the configured conditions are fulfilled or not.

 This attribute makes the internal ConditionMonitor status observable.";

 key id;

 uses top3gpp:Top\_Grp;

 container attributes {

 uses ConditionMonitorGrp;

 }

 }

 }

 augment /me3gpp:ManagedElement {

 if-feature ConditionMonitorUnderManagedElement;

 uses ConditionMonitorSubtree;

 }

 augment /subnet3gpp:SubNetwork {

 if-feature ConditionMonitorUnderSubNetwork;

 uses ConditionMonitorSubtree;

 }

}

<CODE ENDS>

\*\*\* END OF CHANGE 3 \*\*\*

\*\*\* START OF CHANGE 4 \*\*\*

\*\*\* yang-models/\_3gpp-common-yang-types.yang \*\*\*

<CODE BEGINS>

module \_3gpp-common-yang-types {

 yang-version 1.1;

 namespace "urn:3gpp:sa5:\_3gpp-common-yang-types";

 prefix "types3gpp";

 import ietf-inet-types { prefix inet; }

 import ietf-yang-types { prefix yang; }

 import \_3gpp-common-yang-extensions { prefix yext3gpp; }

 organization "3GPP SA5";

 contact "https://www.3gpp.org/DynaReport/TSG-WG--S5--officials.htm?Itemid=464";

 description "The model defines a YANG mapping of the top level

 information classes used for management of 5G networks and

 network slicing.

 Copyright 2024, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI,

 TTA, TTC). All rights reserved.";

 reference "3GPP TS 28.623";

 revision 2024-08-13 { reference CR-0387; }

 revision 2024-05-18 { reference CR-0359; }

 revision 2024-04-12 { reference CR-0346; }

 revision 2023-11-06 { reference CR-0305; }

 revision 2023-09-18 { reference CR-0271 ; }

 revision 2023-08-09 { reference CR-0266; }

 revision 2023-05-10 { reference CR-0250; }

 revision 2023-02-14 { reference CR-0234; }

 revision 2022-11-04 { reference "CR-0194"; }

 revision 2022-10-24 { reference CR-0196; }

 revision 2022-07-26 { reference "CR-0180" ; }

 revision 2022-02-09 { reference "CR-0144"; }

 revision 2021-11-01 { reference "CR-0141"; }

 revision 2021-09-30 {

 description "Added Longitude, Latitude, TenthOfDegrees, OnOff.";

 reference "CR-0138";

 }

 revision 2020-11-06 {

 description "Removed incorrect S-NSSAI definitions.";

 reference "CR-0118";

 }

 revision 2020-03-10 {

 description "Removed faulty when statements.";

 reference "SP-200229";

 }

 revision 2019-10-25 {

 description "Added ManagedNFProfile.";

 reference "S5-194457";

 }

 revision 2019-10-16 {

 description "Added SAP and usageState.";

 reference "S5-193518";

 }

 revision 2019-06-23 {

 reference "Initial version.";

 }

 typedef EnabledDisabled {

 type enumeration {

 enum DISABLED ;

 enum ENABLED ;

 }

 }

 grouping nameValuePair {

 leaf name { type string; }

 leaf value { type string; }

 }

 typedef DayOfWeekT {

 type enumeration {

 enum Monday;

 enum Tuesday;

 enum Wednesday;

 enum Thursday;

 enum Friday;

 enum Saturday;

 enum Sunday;

 }

 }

 typedef DayOfWeek {

 type enumeration {

 enum MONDAY;

 enum TUESDAY;

 enum WEDNESDAY;

 enum THURSADY;

 enum FRIDAY;

 enum SATURDAY;

 enum SUNDAY;

 }

 }

 typedef DateMonth {

 type uint8 {

 range 0..12;

 }

 }

 typedef DateMonthDay {

 type uint8 {

 range 0..31;

 }

 }

 typedef FullTime {

 type yang:time-with-zone-offset;

 }

 grouping DayInYearGrp {

 description "This <<dataType>> represents a day in a year.";

 leaf month {

 description "It indicates the month in a year.";

 type DateMonth;

 mandatory true;

 }

 leaf monthDay {

 description "It indicates the day in a month.";

 type DateMonthDay;

 mandatory true;

 }

 }

 grouping TimeWindowGrp {

 leaf startTime {

 type yang:date-and-time;

 yext3gpp:inVariant;

 }

 leaf endTime {

 type yang:date-and-time;

 yext3gpp:inVariant;

 }

 }

 grouping ProcessMonitorGrp {

 description "Provides attributes to monitor the progress of processes

 with specific purpose and limited lifetime running on MnS producers.

 It may be used as data type for dedicated progress monitor attributes

 when specifying the management representation of these processes.

 The attributes in this clause are defined in a generic way.

 For some attributes specialisations may be provided when specifying a

 concrete process representation.

 If a management operation on some IOCs triggers an associated

 asynchronous process (whose progress shall be monitored), this should

 also result in creating an attribute named 'processMonitor' (of type

 'ProcessMonitor') in these IOC(s). The processMonitor attribute may be

 accompanied by use-case specific additional data items.

 The progress of the process is described by the 'status' and

 'progressPercentage' attributes. Additional textual qualifications for

 the 'status' attribute may be provided by the 'progressStateInfo' and

 'resultStateInfo' attributes.

 When the process is instantiated, the 'status' is set to 'NOT\_RUNNING'

 and the 'progressPercentage' to '0'. The MnS producer decides when to

 start executing the process and to transition into the 'RUNNING' state.

 This time is captured in the 'startTime' attribute. Alternatively, the

 process may start to execute directly upon its instantiation. One

 alternative must be selected when using this data type.

 During the 'RUNNING' state the 'progressPercentage' attribute may be

 repeatedly updated. The exact semantic of this attribute is subject to

 further specialisation. The 'progressInfo' attribute may be used to

 provide additional textual information in the 'NOT\_RUNNING', 'CANCELLING'

 and 'RUNNING' states. Further specialisation of

 'progressStateInfo' may be provided where this data type is

 used.

 Upon successful completion of the process, the 'status' attribute is set

 to 'FINISHED', the 'progressPercentage' to 100%. The time is captured in

 the 'endTime' attribute. Additional textual information may be provided

 in the 'resultStateInfo' attribute. The type of

 'resultStateInfo' in this data type definition is 'String'.

 Further specialisation of 'resultStateInfo' may be provided

 where this data type is used.

 In case the process fails to complete successfully, the 'status'

 attribute is set to 'FAILED' or 'PARTIALLY\_FAILED', the current value of

 'progressPercentage' is frozen, and the time captured in 'endTime'. The

 'resultStateInfo' specifies the reason for the failure.

 Specific failure reasons may be specified where the data type defined in

 this clause is used. The exact semantic of failure may be subject for

 further specialisation as well.

 In case the process is cancelled, the 'status' attribute is first set to

 'CANCELLING' and when the process is really cancelled then to 'CANCELLED'.

 The transition to 'CANCELLED' is captured in the 'endTime' attribute.

 The value of 'progressPercentage' is frozen. Additional textual

 information may be provided in the 'resultStateInfo' attribute.

 The 'resultStateInfo' attribute is provided only for additional textual

 qualification of the states 'FINISHED', 'FAILED', 'PARTIALLY\_FAILED' or

 'CANCELLED'. It shall not be used for making the outcome, that the

 process may produce in case of success, available.

 The process may have to be completed within a certain time after its

 creation, for example because required data may not be available any

 more after a certain time, or the process outcome is needed until a

 certain time and when not provided by this time is not needed any more.

 The time until the MnS producer automatically cancels the process is

 indicated by the 'timer' attribute.";

 leaf id {

 type string;

 mandatory true;

 description "Id of the process. It is unique within a single

 multivalue attribute of type ProcessMonitor.";

 }

 leaf status {

 type enumeration {

 enum NOT\_STARTED ;

 enum RUNNING ;

 enum CANCELLING ;

 enum FINISHED ;

 enum FAILED ;

 enum PARTIALLY\_FAILED ;

 enum CANCELLED ;

 }

 config false;

 default RUNNING;

 description "Represents the status of the associated process,

 whether it fails, succeeds etc.

 It does not represent the returned values of a successfully finished

 process. ";

 }

 leaf progressPercentage {

 type uint8 {

 range 0..100;

 }

 config false;

 description "Progress of the associated process as percentage";

 }

 leaf-list progressStateInfo {

 type string;

 config false;

 description "Additional textual qualification of the states

 'NOT\_STARTED', 'CANCELLING' and 'RUNNING'.

 For specific processes, specific well-defined strings (e.g. string

 patterns or enums) may be defined as a specialisation.";

 }

 leaf resultStateInfo {

 type string;

 config false;

 description "Additional textual qualification of the states

 'FINISHED', 'FAILED', 'PARTIALLY\_FAILED and 'CANCELLED'.

 For example, in the 'FAILED' or 'PARTIALLY\_FAILED' state this

 attribute may be used to provide error reasons.

 This attribute shall not be used to make the outcome of the process

 available for retrieval, if any. For this purpose, dedicated

 attributes shall be specified when specifying the representation of

 a specific process.

 For specific processes, specific well-defined strings (e.g. string

 patterns or enums) may be defined as a specialisation.";

 }

 leaf startTime {

 type yang:date-and-time;

 config false;

 description "Start time of the associated process, i.e. the time when the

 status changed from 'NOT\_STARTED' to 'RUNNING'.";

 }

 leaf endTime {

 type yang:date-and-time;

 config false;

 description "Date and time when status changed to 'SUCCESS', 'CANCELLED',

 'FAILED' or 'PARTIALLY\_FAILED'.

 If the time is in the future, it is the estimated time

 the process will end.";

 }

 leaf timer {

 type uint32;

 units minutes;

 description "Time until the associated process is automatically cancelled.

 If set, the system decreases the timer with time. When it reaches zero

 the cancellation of the associated process is initiated by the

 MnS\_Producer.

 If not set, there is no time limit for the process.

 Once the timer is set, the consumer can not change it anymore.

 If the consumer has not set the timer the MnS Producer may set it.";

 yext3gpp:notNotifyable;

 }

 }

 typedef TenthOfDegrees {

 type uint16 {

 range 0..3600;

 }

 units "0.1 degrees";

 description "A single integral value corresponding to an angle in degrees

 between 0 and 360 with a resolution of 0.1 degrees.";

 }

 typedef Latitude {

 type decimal64 {

 fraction-digits 4;

 range "-90.0000..+90.0000";

 }

 description "Latitude values";

 }

 typedef Longitude {

 type decimal64 {

 fraction-digits 4;

 range "-180.0000..+180.0000";

 }

 description "Longitude values";

 }

 typedef Altitude {

 type decimal64 {

 fraction-digits 6;

 }

 units "meters";

 description

 "Height from a reference 0 value.";

 }

 grouping GeographicalCoordinates {

 description "This datatype represents the geographical coordinates";

 reference "#GPP TS 28.558 clause 6.3.8";

 leaf latitude {

 type Latitude;

 mandatory true;

 }

 leaf longitude {

 type Longitude;

 mandatory true;

 }

 leaf altitude {

 type Altitude;

 }

 }

 typedef OnOff {

 type enumeration {

 enum ON;

 enum OFF;

 }

 }

 // grouping ManagedNFProfile will be removed as it is

 // being moved to \_3gpp-5gc-nrm-nfprofile

 grouping ManagedNFProfile {

 description "Defines profile for managed NF";

 reference "3GPP TS 23.501";

 leaf idx { type uint32 ; }

 leaf nfInstanceID {

 config false;

 mandatory true;

 type yang:uuid ;

 description "This parameter defines profile for managed NF.

 The format of the NF Instance ID shall be a

 Universally Unique Identifier (UUID) version 4,

 as described in IETF RFC 4122 " ;

 yext3gpp:inVariant;

 }

 leaf-list nfType {

 config false;

 min-elements 1;

 type NfType;

 description "Type of the Network Function" ;

 }

 leaf hostAddr {

 mandatory true;

 type inet:host ;

 description "Host address of a NF";

 }

 leaf authzInfo {

 type string ;

 description "This parameter defines NF Specific Service authorization

 information. It shall include the NF type (s) and NF realms/origins

 allowed to consume NF Service(s) of NF Service Producer.";

 reference "See TS 23.501" ;

 }

 leaf location {

 type string ;

 description "Information about the location of the NF instance

 (e.g. geographic location, data center) defined by operator";

 reference "TS 29.510" ;

 }

 leaf capacity {

 mandatory true;

 type uint16 ;

 description "This parameter defines static capacity information

 in the range of 0-65535, expressed as a weight relative to other

 NF instances of the same type; if capacity is also present in the

 nfServiceList parameters, those will have precedence over this value.";

 reference "TS 29.510" ;

 }

 leaf nFSrvGroupId {

 type string ;

 description "This parameter defines identity of the group that is

 served by the NF instance.

 May be config false or true depending on the ManagedFunction.

 Config=true for Udrinfo. Config=false for UdmInfo and AusfInfo.

 Shall be present if ../nfType = UDM or AUSF or UDR. ";

 reference "TS 29.510" ;

 }

 leaf-list supportedDataSetIds {

 type enumeration {

 enum SUBSCRIPTION;

 enum POLICY;

 enum EXPOSURE;

 enum APPLICATION;

 }

 description "List of supported data sets in the UDR instance.

 May be present if ../nfType = UDR";

 reference "TS 29.510" ;

 }

 leaf-list smfServingAreas {

 type string ;

 description "Defines the SMF service area(s) the UPF can serve.

 Shall be present if ../nfType = UPF";

 reference "TS 29.510" ;

 }

 leaf priority {

 type uint16;

 description "This parameter defines Priority (relative to other NFs

 of the same type) in the range of 0-65535, to be used for NF selection;

 lower values indicate a higher priority. If priority is also present

 in the nfServiceList parameters, those will have precedence over

 this value. Shall be present if ../nfType = AMF ";

 reference "TS 29.510" ;

 }

 }

 typedef usageState {

 type enumeration {

 enum IDLE;

 enum ACTIVE;

 enum BUSY;

 }

 description "It describes whether or not the resource is actively in

 use at a specific instant, and if so, whether or not it has spare

 capacity for additional users at that instant. The value is READ-ONLY.";

 reference "ITU T Recommendation X.731";

 }

 grouping SAPGrp {

 leaf host {

 type inet:host;

 mandatory true;

 }

 leaf port {

 type inet:port-number;

 mandatory true;

 }

 description "Service access point.";

 reference "TS 28.622";

 }

 typedef Mcc {

 description "The mobile country code consists of three decimal digits,

 The first digit of the mobile country code identifies the geographic

 region (the digits 1 and 8 are not used):";

 type string {

 pattern '[02-79][0-9][0-9]';

 }

 reference "3GPP TS 23.003 subclause 2.2 and 12.1";

 }

 typedef Mnc {

 description "The mobile network code consists of two or three

 decimal digits (for example: MNC of 001 is not the same as MNC of 01)";

 type string {

 pattern '[0-9][0-9][0-9]|[0-9][0-9]';

 }

 reference "3GPP TS 23.003 subclause 2.2 and 12.1";

 }

 grouping PLMNId {

 leaf mcc {

 mandatory true;

 type Mcc;

 }

 leaf mnc {

 mandatory true;

 type Mnc;

 }

 reference "TS 38.413 clause 9.3.3.5";

 }

 typedef Nci {

 description "NR Cell Identity. The NCI shall be of fixed length of 36 bits

 and shall be coded using full hexadecimal representation.

 The exact coding of the NCI is the responsibility of each PLMN operator";

 reference "TS 23.003";

 type union {

 type string {

 length 36;

 pattern '[01]+';

 }

 type string {

 length 9;

 pattern '[a-fA-F0-9]\*';

 }

 }

 }

 typedef OperationalState {

 reference "3GPP TS 28.625 and ITU-T X.731";

 type enumeration {

 enum DISABLED {

 value 0;

 description "The resource is totally inoperable.";

 }

 enum ENABLED {

 value 1;

 description "The resource is partially or fully operable.";

 }

 }

 }

 typedef BasicAdministrativeState {

 reference "3GPP TS 28.625 and ITU-T X.731";

 type enumeration {

 enum LOCKED {

 value 0;

 description "The resource is administratively prohibited from performing

 services for its users.";

 }

 enum UNLOCKED {

 value 1;

 description "The resource is administratively permitted to perform

 services for its users. This is independent of its inherent

 operability.";

 }

 }

 }

 typedef AdministrativeState {

 reference "3GPP TS 28.625 and ITU-T X.731";

 type enumeration {

 enum LOCKED {

 value 0;

 description "The resource is administratively prohibited from performing

 services for its users.";

 }

 enum UNLOCKED {

 value 1;

 description "The resource is administratively permitted to perform

 services for its users. This is independent of its inherent

 operability.";

 }

 enum SHUTTINGDOWN {

 value 2;

 description "Use of the resource is administratively permitted to

 existing instances of use only. While the system remains in

 the shutting down state the manager or the managed element

 may at any time cause the resource to transition to the

 locked state.";

 }

 }

 }

 typedef AvailabilityStatus {

 type enumeration {

 enum IN\_TEST;

 enum FAILED;

 enum POWER\_OFF;

 enum OFF\_LINE;

 enum OFF\_DUTY;

 enum DEPENDENCY;

 enum DEGRADED;

 enum NOT\_INSTALLED;

 enum LOG\_FULL;

 }

 }

 typedef CellState {

 type enumeration {

 enum IDLE;

 enum INACTIVE;

 enum ACTIVE;

 }

 }

 typedef Nrpci {

 type uint32;

 description "Physical Cell Identity (PCI) of the NR cell.";

 reference "TS 36.211 subclause 6.11";

 }

 typedef Tac {

 type int32 {

 range 0..16777215 ;

 }

 description "Tracking Area Code";

 reference "TS 23.003 clause 19.4.2.3";

 }

 grouping TaiGrp {

 description "This <<dataType>> defines a Tracking Area Identity (TAI)

 as specified in clause 28.6 of TS 23.003, clause 8.2 of TS 38.300

 and clause 9.3.3.11 of TS 38.413. It is composed of the PLMN

 identifier (PLMN-Id, which is composed of the MCC and MNC) and

 the Tracking Area Code (TAC). ";

 list plmnId {

 description "PLMN Identity.";

 min-elements 1;

 max-elements 1;

 key "mcc mnc";

 uses types3gpp:PLMNId;

 }

 leaf tac { type Tac; }

 }

 grouping GeoCoordinateGrp {

 description "Geographical location on earth";

 leaf latitude {

 type decimal64 {

 fraction-digits 4;

 range -90..90 ;

 }

 mandatory true;

 description "Latitude based on World Geodetic System (1984 version)

 global reference frame (WGS 84). Positive values correspond to the

 northern hemisphere.";

 }

 leaf longitude {

 type decimal64 {

 fraction-digits 4;

 range -180..180 ;

 }

 mandatory true;

 description "Longitude based on World Geodetic System (1984 version)

 global reference frame (WGS 84). Positive values correspond to

 degrees east of 0 degrees longitude.";

 }

 }

 grouping GeoAreaGrp {

 description "This data type defines a geographical area.

 The geo-area is defined using a convex polygon in the attribute

 'convexGeoPolygon'.";

 list convexGeoPolygon {

 description "Specifies the geographical area with a convex polygon.

 The convex polygon is specified by its corners.";

 key "latitude longitude";

 min-elements 3;

 ordered-by user;

 uses GeoCoordinateGrp;

 }

 }

 typedef AmfRegionId {

 type union {

 type uint8 ;

 type string {

 length 8;

 pattern '[01]\*';

 }

 }

 reference "clause 2.10.1 of 3GPP TS 23.003";

 }

 typedef AmfSetId {

 type union {

 type uint16 {

 range '0..1023';

 }

 type string {

 length 8;

 pattern '[01]\*';

 }

 }

 reference "clause 2.10.1 of 3GPP TS 23.003";

 }

 typedef AmfPointer {

 type union {

 type uint8 {

 range '0..63';

 }

 type string {

 length 6;

 pattern '[01]\*';

 }

 }

 reference "clause 2.10.1 of 3GPP TS 23.003";

 }

 grouping AmfIdentifier {

 leaf amfRegionId {

 type AmfRegionId;

 }

 leaf amfSetId {

 type AmfSetId;

 }

 leaf amfPointer {

 type AmfPointer;

 }

 description "The AMFI is constructed from an AMF Region ID,

 an AMF Set ID and an AMF Pointer.

 The AMF Region ID identifies the region,

 the AMF Set ID uniquely identifies the AMF Set within the AMF Region, and

 the AMF Pointer uniquely identifies the AMF within the AMF Set. ";

 }

// type definitions especially for core NFs

 typedef NfType {

 type enumeration {

 enum NRF;

 enum UDM;

 enum AMF;

 enum SMF;

 enum AUSF;

 enum NEF;

 enum PCF;

 enum SMSF;

 enum NSSF;

 enum UDR;

 enum LMF;

 enum GMLC;

 enum 5G\_EIR;

 enum SEPP;

 enum UPF;

 enum N3IWF;

 enum AF;

 enum UDSF;

 enum BSF;

 enum CHF;

 }

 }

 typedef NotificationType {

 type enumeration {

 enum N1\_MESSAGES;

 enum N2\_INFORMATION;

 enum LOCATION\_NOTIFICATION;

 }

 }

 typedef Load {

 description "Latest known load information of the NF, percentage ";

 type uint8 {

 range 0..100;

 }

 }

 typedef N1MessageClass {

 type enumeration {

 enum 5GMM;

 enum SM;

 enum LPP;

 enum SMS;

 }

 }

 typedef N2InformationClass {

 type enumeration {

 enum SM;

 enum NRPPA;

 enum PWS;

 enum PWS\_BCAL;

 enum PWS\_RF;

 }

 }

 grouping DefaultNotificationSubscription {

 leaf notificationType {

 type NotificationType;

 }

 leaf callbackUri {

 type inet:uri;

 }

 leaf n1MessageClass {

 type N1MessageClass;

 }

 leaf n2InformationClass {

 type N2InformationClass;

 }

 }

 grouping Ipv4AddressRange {

 leaf start {

 type inet:ipv4-address;

 }

 leaf end {

 type inet:ipv4-address;

 }

 }

 grouping Ipv6PrefixRange {

 leaf start {

 type inet:ipv6-prefix;

 }

 leaf end {

 type inet:ipv6-prefix;

 }

 }

 typedef NsiId {

 type string;

 }

 typedef UeMobilityLevel {

 type enumeration {

 enum STATIONARY;

 enum NOMADIC;

 enum RESTRICTED\_MOBILITY;

 enum FULLY\_MOBILITY;

 }

 }

 typedef ResourceSharingLevel {

 type enumeration {

 enum SHARED;

 enum NOT\_SHARED;

 }

 }

 typedef TxDirection {

 type enumeration {

 enum DL;

 enum UL;

 enum DL\_AND\_UL;

 }

 }

 grouping AddressWithVlan {

 leaf ipAddress {

 type inet:ip-address;

 }

 leaf vlanId {

 type uint16;

 }

 }

 /\* DistinguishedName pattern is built up based on the

 EBNF in 32.300 clause 7.3 EBNF of DN String Representation

 leaf DN { type string { // Same pattern as LocalDN

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?(,[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?)\*';

 } }

 leaf fullLocalDN { type string { // LocalRDN , { RDNSeparator , LocalRDN } RDNSeparator is a single , no space or \R allowed Me.mykey=1 allowed

 // (fullLocalRDN)(,(fullLocalRDN))\*

 pattern '(([A-Z][^,=+<>#;\\"\r\n\*.]\*|([A-Z][^,=+<>#;\\"\r\n\*.]\*\.[a-z][^,=+<>#;\\"\r\n\*.]\*))=(([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?))(,(([A-Z][^,=+<>#;\\"\r\n\*.]\*|([A-Z][^,=+<>#;\\"\r\n\*.]\*\.[a-z][^,=+<>#;\\"\r\n\*.]\*))=(([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?)))\*';

 } }

 leaf LocalDN { type string { // LocalRDN , { RDNSeparator , LocalRDN } RDNSeparator is a single , no space or \R allowed

 // LocalRDN(,LocalRDN)\*

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?(,[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?)\*';

 } }

 leaf fullLocalRDN { type string { // same as fullLocalDNAttributeTypeAndValue

 pattern '([A-Z][^,=+<>#;\\"\r\n\*.]\*|([A-Z][^,=+<>#;\\"\r\n\*.]\*\.[a-z][^,=+<>#;\\"\r\n\*.]\*))=(([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?)';

 } }

 leaf LocalRDN { type string { // same as LocalDNAttributeTypeAndValue

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?';

 } }

 leaf fullLocalDNAttributeTypeAndValue { type string { // LocalDNAttributeType , AttributeTypeAndValueSeparator , RegularAttributeValue

 // pattern LocalDNAttributeType=RegularAttributeValue

 pattern '([A-Z][^,=+<>#;\\"\r\n\*.]\*|([A-Z][^,=+<>#;\\"\r\n\*.]\*\.[a-z][^,=+<>#;\\"\r\n\*.]\*))=(([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?)';

 } }

 // limitation: NamesOfClassAndNamingAttributenot supported Me.mykey=1

 leaf LocalDNAttributeTypeAndValue { type string {

 // ebnf1 LocalDNAttributeType , AttributeTypeAndValueSeparator , RegularAttributeValue

 // ebnf2-limited NameOfClassWithIdAttribute , AttributeTypeAndValueSeparator , RegularAttributeValue

 // pattern NameOfClassWithIdAttribute=RegularAttributeValue

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?';

 } }

 leaf LocalDNAttributeType { type string { // NameOfClassWithIdAttribute | NamesOfClassAndNamingAttribute RDNSeparator is a single , no space or \R allowed

 // NameOfClassWithIdAttribute|NamesOfClassAndNamingAttribute

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*|([A-Z][^,=+<>#;\\"\r\n\*.]\*\.[a-z][^,=+<>#;\\"\r\n\*.]\*)';

 } }

 leaf RegularAttributeValue { type string { // ( AttributeValueChar - SpaceChar ) , [ { AttributeValueChar } , ( AttributeValueChar - SpaceChar ) ]

 pattern '([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?' ;

 } }

 leaf NamesOfClassAndNamingAttribute { type string { // ClassName , ClassNamingAttributeSeparator , NamingAttributeName

 // pattern: ClassName\.NamingAttributeName

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*\.[a-z][^,=+<>#;\\"\r\n\*.]\*' ;

 } }

 leaf restrictiveClassName { type string { //

 pattern '[a-zA-Z][a-zA-Z0-9-\_]\*' ;

 } }

 leaf ClassName { type string { // CapitalLetterChar , { LocalDNAttributeTypeChar }

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*' ;

 } }

 leaf NamingAttributeName { type string { // SmallLetterChar , { LocalDNAttributeTypeChar }

 pattern '[a-z][^,=+<>#;\\"\r\n\*.]\*' ;

 } }

 \*/

 typedef DistinguishedName {

 type string {

 pattern '[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|'

 + '(\\[a-fA-F0-9]{2}))(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*'

 + '([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?'

 + '(,[A-Z][^,=+<>#;\\"\r\n\*.]\*=([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2}))'

 + '(([^,=+<>#;\\"\r\n\*]|(\\[a-fA-F0-9]{2}))\*'

 + '([^,=+<>#;\\"\r\n\* ]|(\\[a-fA-F0-9]{2})))?)\*';

 }

 description "Represents the 3GPP standard for DistinguishedName.

 Limitations:

 - RDNSeparator: don't allow SpaceChar or CarriageReturnChar

 - NullDN: Disallow nullDN that is the same as not providing a DN

 - NamesOfClassAndNamingAttribute format not allowed

 (eg. ManagedElement.mykey=345436)";

 reference "3GPP TS 32.300";

 }

 typedef QOffsetRange {

 type int8 {

 range "-24 | -22 | -20 | -18 | -16 | -14 | -12 | -10 | -8 | -6 | " +

 " -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 8 | 10 | " +

 " 12 | 14 | 16 | 18 | 20 | 22 | 24";

 }

 units dB;

 }

 grouping ReportingCtrl {

 choice reportingCtrl {

 mandatory true;

 description "

 This choice defines the method for reporting collected performance

 metrics to MnS consumers as well as the parameters for configuring the

 reporting function. It is a choice between the control parameter

 required for the reporting methods, whose presence selects the

 reporting method as follows:

 - When only the fileReportingPeriod attribute is present, the MnS

 producer shall store files on the MnS producer at a location selected

 by the MnS producer and, on condition that an appropriate subscription

 is in place, inform the MnS consumer about the availability of new

 files and the file location using the notifyFileReady notification.

 In case the preparation of a file fails, 'notifyFilePreparationError'

 shall be sent instead.

 - When the 'fileReportingPeriod' and 'notificationRecipientAddress'

 attributes are present, then the MnS producer shall behave like

 described for the case that only the 'fileReportingPeriod' is present.

 In addition, the MnS producer shall create on behalf of the MnS

 consumer a subscription, using 'NtfSubscriptionControl', for the

 notification types 'notifyMOICreation' and 'notifyMOIDeletion' related

 to the 'File' instances that will be produced later. In case an existing

 subscription does already include the 'File' instances to be produced,

 no new subscription shall be created. The

 'notificationRecipientAddress' attribute in the created

 'NtfSubscriptionControl' instance shall be set to the value of the

 'notificationRecipientAddress' in the related 'PerfMetricJob'. This

 feature is called implicit notification subscription, as opposed to the

 case where the MnS consumer creates the subscription (explicit

 notification subscription). When the related 'PerfMetricJob' is

 deleted, the 'NtfSubscriptionControl' instance created due to the

 request for implicit subscription shall be deleted as well.

 - When only the fileReportingPeriod and fileLocation attributes are

 present, the MnS producer shall store the files on a MnS consumer, that

 can be any entity such as a file server, at the location specified by

 fileLocation. No notification is emitted by the MnS producer.

 - When only the streamTarget attribute is present, the MnS producer

 shall stream the data to the location specified by streamTarget.

 For the file-based reporting methods the fileReportingPeriod attribute

 specifies the time window during which collected measurements are

 stored into the same file before the file is closed and a new file is

 opened.";

 case file-based-reporting {

 leaf fileReportingPeriod {

 type uint32 {

 range 1..max;

 }

 units minutes;

 mandatory true;

 description "For the file-based reporting method this is the time

 window during which collected measurements are stored into the same

 file before the file is closed and a new file is opened.

 The time-period must be a multiple of the granularityPeriod.

 Applicable when the file-based reporting method is supported.";

 }

 choice reporting-target {

 case file-target {

 leaf fileLocation {

 type string ;

 description "Applicable and must be present when the file-based

 reporting method is supported, and the files are stored on the MnS

 consumer.";

 }

 }

 case notification-target {

 leaf notificationRecipientAddress {

 type string;

 description "Must be present when the notification-based reporting

 method is supported, and the the files are available as

 notifications for the MnS consumer to subscribe to.";

 }

 }

 description "When netiher fileLocation or notificationRecipientAddress

 are present, the files are stored and available to the MnS consumer

 if the MnS subscribes to the notifyFileReady notification.";

 }

 }

 case stream-based-reporting {

 leaf streamTarget {

 type string;

 mandatory true;

 description "Applicable when stream-based reporting method is

 supported.";

 }

 }

 }

 }

}

<CODE ENDS>

\*\*\* END OF CHANGE 4 \*\*\*