**3GPP TSG-SA5 Meeting #154 *S5-242032***

Changsha, China, 15 - 19 April 2024

|  |
| --- |
| *CR-Form-v12.1* |
| **CHANGE REQUEST** |
|  |
|  | **28.623** | **CR** |  | **rev** | **0347** | **Current version:** | **17.9.0** |  |
|  |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* |
|  |

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

|  |
| --- |
|  |
| ***Title:***  |  |
|  |  |
| ***Source to WG:*** | Ericsson |
| ***Source to TSG:*** | S5 |
|  |  |
| ***Work item code:*** | TEI17 |  | ***Date:*** | 2024-04-16 |
|  |  |  |  |  |
| ***Category:*** | F |  | ***Release:*** | Rel-17 |
|  | *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-15 (Release 15)Rel-16 (Release 16)Rel-17 (Release 17)Rel-18 (Release 18)* |
|  |  |
| ***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:*** | D.2.3, D.2.4, D.2.9 |
|  |  |
|  | **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:*** | YANG Forge link: Forge MR link: <https://forge.3gpp.org/rep/sa5/MnS/-/merge_requests/1105> at commit 021cf1a2b5054d9d9fc1915427c16cce1f0d5d1e |
|  |  |
| ***This CR's revision history:*** |  |

***First change***

## D.2.3 module \_3gpp-common-managed-function.yang

<CODE BEGINS>

module \_3gpp-common-managed-function {

 yang-version 1.1;

 namespace urn:3gpp:sa5:\_3gpp-common-managed-function;

 prefix mf3gpp;

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

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

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

 import \_3gpp-common-measurements { prefix meas3gpp; }

 import \_3gpp-common-trace { prefix trace3gpp; }

 organization "3GPP SA5";

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

 description "The module defines a base class/grouping for major 3GPP

 functions.

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

 TTA, TTC). All rights reserved.";

 reference

 "3GPP TS 28.622

 Generic Network Resource Model (NRM)

 Integration Reference Point (IRP);

 Information Service (IS)

 3GPP TS 28.620

 Umbrella Information Model (UIM)";

 revision 2024-02-23 { reference CR-0347 ; }

 revision 2023-12-17 { reference CR-0327 ; }

 revision 2023-09-17 { reference CR-0270 ; }

 revision 2022-10-31 { reference CR-0195; }

 revision 2022-01-07 { reference "CR-0146"; }

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

 revision 2020-09-30 { reference "CR-bbbb"; }

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

 revision 2020-08-03 { reference "CR-0095"; }

 revision 2020-06-23 { reference "CR-085"; }

 revision 2020-06-08 { reference "CR-0092"; }

 revision 2019-11-21 { reference "S5-197275, S5-197735"; }

 revision 2019-10-28 { reference S5-193518 ; }

 revision 2019-06-18 { reference "Initial revision"; }

 feature MeasurementsUnderManagedFunction {

 description "The MeasurementSubtree shall be contained under ManageElement";

 }

 feature TraceUnderManagedFunction {

 description "The TraceSubtree shall be contained under ManagedFunction";

 }

 grouping OperationGrp {

 description "This data type represents an Operation.";

 reference "3gpp TS 28.622";

 leaf name {

 type string;

 mandatory true;

 yext3gpp:notNotifyable;

 yext3gpp:inVariant;

 }

 leaf-list allowedNFTypes {

 type string;

 min-elements 1;

 description "The type of the managed NF service instance

 The specifc values allowed are described in TS 23.501";

 }

 leaf operationSemantics {

 type enumeration {

 enum REQUEST\_RESPONSE;

 enum SUBSCRIBE\_NOTIFY;

 }

 config false;

 mandatory true;

 description "Semantics type of the operation.";

 reference "3GPP TS 23.502";

 yext3gpp:inVariant;

 }

 }

 grouping ManagedNFServiceGrp {

 description "A ManagedNFService represents a Network Function (NF) service.";

 reference "Clause 7 of 3GPP TS 23.501.";

 leaf userLabel {

 type string;

 description "A user-friendly (and user assignable) name of this object.";

 }

 leaf nFServiceType {

 config false;

 mandatory true;

 type string;

 description "The type of the managed NF service instance

 The specifc values allowed are described in clause 7.2 of TS 23.501";

 yext3gpp:notNotifyable;

 yext3gpp:inVariant;

 }

 list sAP {

 key "host port";

 min-elements 1;

 max-elements 1;

 description "The service access point of the managed NF service instance";

 uses types3gpp:SAPGrp;

 }

 list operations {

 key name;

 min-elements 1;

 uses OperationGrp ;

 description "Set of operations supported by the managed NF

 service instance";

 }

 leaf administrativeState {

 type types3gpp:BasicAdministrativeState;

 mandatory true;

 description "Permission to use or prohibition against using the instance";

 }

 leaf operationalState {

 type types3gpp:OperationalState;

 config false;

 mandatory true;

 description "Describes whether the resource is installed and working";

 }

 leaf usageState {

 type types3gpp:usageState ;

 config false;

 mandatory true;

 description "Describes whether the resource is actively in use at a

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

 capacity for additional users.";

 }

 leaf registrationState {

 type enumeration {

 enum REGISTERED;

 enum DEREGISTERED;

 }

 config false;

}

 }

 grouping Function\_Grp {

 description "A base grouping for 3GPP functions.";

 leaf userLabel {

 type string;

 description "A user-friendly (and user assignable) name of this object.";

 }

 }

 grouping ManagedFunctionGrp {

 description "Abstract root class to be inherited/reused by classes

 representing 3GPP functions.

 Anywhere this grouping is used by classes inheriting from ManagedFunction

 the list representing the inheriting class needs to include all

 contained classes of ManagedFunction too. Contained classes are

 either

 - augmented into the Function class or

 - shall be included in the list representing the inheriting class

 using the grouping ManagedFunctionContainedClasses:

 1) EP\_RP solved using augment

 2) uses mf3gpp:ManagedFunctionContainedClasses;

 ";

 uses Function\_Grp;

 list vnfParametersList {

 key vnfInstanceId;

 description "Contains the parameter set of the VNF

 instance(s) corresponding to an NE.

 The presence of this list indicates that the ManagedFunction

 represented is realized by one or more VNF instance(s). Otherwise it

 shall be absent.

 The presence of a vnfParametersList entry, whose vnfInstanceId with a

 string length of zero, in createMO operation can trigger the

 instantiation of the related VNF/VNFC instances.";

 leaf vnfInstanceId {

 type string ;

 description "VNF instance identifier";

 reference "ETSI GS NFV-IFA 008 v2.1.1:

 Network Functions Virtualisation (NFV); Management and Orchestration;

 Ve-Vnfm reference point - Interface and Information Model Specification

 section 9.4.2

 ETSI GS NFV-IFA 015 v2.1.2: Network Functions Virtualisation (NFV);

 Management and Orchestration; Report on NFV Information Model

 section B2.4.2.1.2.3";

 }

 leaf vnfdId {

 type string ;

 description "Identifier of the VNFD on which the VNF instance is based.

 The absence of the leaf or a string length of zero for vnfInstanceId

 means the VNF instance(s) does not exist (e.g. has not been

 instantiated yet, has already been terminated).";

 reference "ETSI GS NFV-IFA 008 v2.1.1:

 Network Functions Virtualisation (NFV); Management and Orchestration;

 Ve-Vnfm reference point - Interface and Information Model Specification

 section 9.4.2";

 }

 leaf flavourId {

 type string ;

 description "Identifier of the VNF Deployment Flavour applied to this

 VNF instance.";

 reference "ETSI GS NFV-IFA 008 v2.1.1:

 Network Functions Virtualisation (NFV) Management and Orchestration";

 }

 leaf autoScalable {

 type boolean ;

 mandatory true;

 description "Indicator of whether the auto-scaling of this

 VNF instance is enabled or disabled.";

 }

 }

 list peeParametersList {

 key idx;

 description "Contains the parameter set for the control

 and monitoring of power, energy and environmental parameters of

 ManagedFunction instance(s).";

 leaf idx { type uint32; }

 leaf siteIdentification {

 type string;

 mandatory true;

 description "The identification of the site where the

 ManagedFunction resides.";

 }

 leaf siteLatitude {

 type decimal64 {

 fraction-digits 4;

 range "-90.0000..+90.0000";

 }

 description "The latitude of the site where the ManagedFunction

 instance resides, based on World Geodetic System (1984 version)

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

 the northern hemisphere. This attribute is optional

 for BTSFunction, RNCFunction, GNBDUFunction and

 NRSectorCarrier instance(s).";

 }

 leaf siteLongitude {

 type decimal64 {

 fraction-digits 4;

 range "-180.0000..+180.0000";

 }

 description "The longitude of the site where the ManagedFunction

 instance resides, based on World Geodetic System (1984 version)

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

 degrees east of 0 degrees longitude. This attribute is optional

 for BTSFunction, RNCFunction, GNBDUFunction and

 NRSectorCarrier

 instance(s).";

 }

 leaf siteAltitude {

 type decimal64 {

 fraction-digits 4;

 }

 description "The altitude of the site where the ManagedFunction

 instance resides, in the unit of meter. This attribute is

 optional for BTSFunction, RNCFunction, GNBDUFunction and

 NRSectorCarrier instance(s).";

 }

 leaf siteDescription {

 type string;

 mandatory true;

 description "An operator defined description of the site where

 the ManagedFunction instance resides.";

 }

 leaf equipmentType {

 type string;

 mandatory true;

 description "The type of equipment where the managedFunction

 instance resides.";

 reference "clause 4.4.1 of ETSI ES 202 336-12";

 }

 leaf environmentType {

 type string;

 mandatory true;

 description "The type of environment where the managedFunction

 instance resides.";

 reference "clause 4.4.1 of ETSI ES 202 336-12";

 }

 leaf powerInterface {

 type string;

 mandatory true;

 description "The type of power.";

 reference "clause 4.4.1 of ETSI ES 202 336-12";

 }

 }

 leaf priorityLabel {

 mandatory true;

 type uint32;

 }

 uses meas3gpp:SupportedPerfMetricGroupGrp;

 }

 grouping ManagedFunctionContainedClasses {

 description "A grouping used to containe classes (lists) contained by

 the abstract IOC ManagedFunction";

 list ManagedNFService {

 description "Represents a Network Function (NF)";

 reference "3GPP TS 23.501";

 key id;

 uses top3gpp:Top\_Grp;

 container attributes {

 uses ManagedNFServiceGrp;

 }

 }

 uses meas3gpp:MeasurementSubtree {

 if-feature MeasurementsUnderManagedFunction ;

 }

 uses trace3gpp:TraceSubtree {

 if-feature TraceUnderManagedFunction ;

 }

 }

}

<CODE ENDS>

D.2.4 module \_3gpp-common-measurements.yang

<CODE BEGINS>

module \_3gpp-common-measurements {

 yang-version 1.1;

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

 prefix "meas3gpp";

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

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

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

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

 organization "3GPP SA5";

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

 description "Defines Measurement and KPI related groupings

 Any list/class intending to use this should include 2 or 3 uses statements

 controlled by a feature:

 A)

+++ feature MeasurementsUnderMyClass {

+++ description 'Indicates whether measurements and/or KPIs are supported

+++ for this class.';

+++ }

 B) include the attribute measurementsList and/or kPIsList indicating the

 supported measurment and KPI types and GPs. Note that for classes

 inheriting from ManagedFunction, EP\_RP or SubNetwork these attributes are

 already inherited, so there is no need to include them once more. E.g.

+++ grouping MyClassGrp {

+++ uses meas3gpp:SupportedPerfMetricGroup;

+++ }

 C) include the class PerfmetricJob to control the measurements/KPIs. E.g.

 list MyClass {

 container attributes {

 uses MyClassGrp;

 }

+++ uses meas3gpp:MeasurementSubtree {

+++ if-feature MeasurementsUnderMyClass ;

+++ }

 }

 Measurements can be contained under ManagedElement, SubNetwork, or

 any list representing a class inheriting from Subnetwork or

 ManagedFunction. Note: KPIs will only be supported under SubNetwork

 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 2024-02-23 { reference CR-0347; }

 revision 2024-01-29 { reference "CR-0327"; }

 revision 2023-11-17 { reference "CR-0298"; }

 revision 2023-11-05 { reference CR-0306; }

 revision 2023-09-17 { reference CR-0270 ; }

 revision 2023-02-15 { reference "CR-0239";}

 revision 2022-11-03 { reference "CR-0193"; }

 revision 2022-10-31 { reference CR-0195; }

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

 revision 2020-11-06 { reference "CR-0118"; }

 revision 2020-09-04 { reference "CR-000107"; }

 revision 2020-06-08 { reference "CR-0092"; }

 revision 2020-05-31 { reference "CR-0084"; }

 revision 2020-03-11 { reference "S5-201581, SP-200229"; }

 revision 2019-11-21 { reference "S5-197275, S5-197735"; }

 revision 2019-10-28 { reference "S5-193516"; }

 revision 2019-06-17 { reference " "; }

 feature FilesUnderPerfMetricJob {

 description "Files shall be contained under PerfMetricJob";

 }

 grouping ThresholdInfoGrp {

 description "Defines a single threshold level.";

 leaf-list performanceMetrics {

 type string;

 min-elements 1;

 description "List of performance metrics.

 Performance metrics include measurements defined in TS 28.552 and KPIs

 defined in TS 28.554 [28]. Performance metrics can also be specified

 by other SDOs, or be vendor specific. Performance metrics are

 identified with their names.

 For measurements defined in TS 28.552 the name is constructed as follows:

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

 subcounters

 - 'family.measurementName' for measurement types without subcounters

 `- 'family' for measurement families

 For KPIs defined in TS 28.554 [28] the name is defined in the KPI

 definitions template as the component designated with e).

 A name can also identify a vendor specific performance metric or a

 group of vendor specific performance metrics.";

 }

 leaf thresholdDirection {

 type enumeration {

 enum UP;

 enum DOWN;

 enum UP\_AND\_DOWN;

 }

 must '. = "UP\_AND\_DOWN" or not(../hysteresis)' {

 error-message "In case a threshold with hysteresis is configured, the "

 +"threshold direction attribute shall be set to 'UP\_AND\_DOWN'.";

 }

 mandatory true;

 description "Direction of a threshold indicating the direction for which

 a threshold crossing triggers a threshold.

 When the threshold direction is configured to 'UP', the associated

 treshold is triggered only when the performance metric value is going

 up upon reaching or crossing the threshold value. The treshold is not

 triggered, when the performance metric is going down upon reaching or

 crossing the threshold value.

 Vice versa, when the threshold direction is configured to 'DOWN', the

 associated treshold is triggered only when the performance metric is

 going down upon reaching or crossing the threshold value. The treshold

 is not triggered, when the performance metric is going up upon reaching

 or crossing the threshold value.

 When the threshold direction is set to 'UP\_AND\_DOWN' the treshold is

 active in both direcions.

 In case a threshold with hysteresis is configured, the threshold

 direction attribute shall be set to 'UP\_AND\_DOWN'.";

 }

 leaf thresholdValue {

 type union {

 type int64;

 type decimal64 {

 fraction-digits 2;

 }

 }

 mandatory true;

 description "Value against which the monitored performance metric is

 compared at a threshold level in case the hysteresis is zero";

 }

 leaf hysteresis {

 type union {

 type uint64;

 type decimal64 {

 fraction-digits 2;

 range "0..max";

 }

 }

 must '. >= 0';

 description "Hysteresis of a threshold. If this attribute is present

 the monitored performance metric is not compared against the

 threshold value as specified by the thresholdValue attribute but

 against a high and low threshold value given by

 threshold-high = thresholdValue + hysteresis

 threshold-low = thresholdValue - hysteresis

 When going up, the threshold is triggered when the performance metric

 reaches or crosses the high threshold value. When going down, the

 hreshold is triggered when the performance metric reaches or crosses

 the low threshold value.

 A hysteresis may be present only when the monitored performance

 metric is not of type counter that can go up only. If present

 for a performance metric of type counter, it shall be ignored.";

 }

 }

 grouping SupportedPerfMetricGroupGrp {

 list SupportedPerfMetricGroups {

 config false;

 description "Captures a group of supported performance metrics and

 associated parameters related to their production and reporting.

 A SupportedPerfMetricGroup attribute which is part of an MOI may

 define performanceMetrics for any MOI under the subtree contained

 under that MOI, e.g. SupportedPerfMetricGroup on a ManagedElement

 can specify supported metrics for contained ManagedFunctions

 like a GNBDUFunction.";

 leaf-list performanceMetrics {

 type string;

 min-elements 1;

 description "Performance metrics include measurements defined in

 TS 28.552 and KPIs defined in TS 28.554.

 Performance metrics are identified by name.

 For measurements defined in TS 28.552 the name is constructed as

 follows:

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

 subcounters

 - 'family.measurementName' for measurement types without subcounters

 - 'family' for measurement families

 For KPIs defined in TS 28.554 the name is defined in the KPI

 definitions template as the component designated with e).

 For non-3GPP specified measurements the name is defined

 elsewhere.";

 }

 leaf-list granularityPeriods {

 type uint32 {

 range 1..max ;

 }

 units seconds;

 description "Granularity periods supported for the associated

 measurement types. The period is defined in seconds.";

 }

 leaf-list reportingMethods {

 type enumeration {

 enum FILE\_BASED\_LOC\_SET\_BY\_PRODUCER;

 enum FILE\_BASED\_LOC\_SET\_BY\_CONSUMER;

 enum STREAM\_BASED;

 }

 min-elements 1;

 }

 leaf-list reportingPeriods {

 type uint32 {

 range 1..max ;

 }

 units seconds;

 description "Reporting periods supported for the associated

 measurement types. The period is defined in seconds.";

 }

 }

 }

 grouping PerfMetricJobGrp {

 description "Represents the attributtes of the IOC PerfMetricJob";

 leaf administrativeState {

 default UNLOCKED;

 type types3gpp:BasicAdministrativeState ;

 description "Enable or disables production of the metrics";

 }

 leaf operationalState {

 config false;

 mandatory true;

 type types3gpp:OperationalState ;

 description "Indicates whether the PerfMetricJob is working.";

 }

 leaf jobId {

 type string;

 description "Id for a PerfMetricJob job.";

 yext3gpp:inVariant;

 }

 leaf-list performanceMetrics {

 type string;

 min-elements 1;

 description "Performance metrics include measurements defined in

 TS 28.552 and KPIs defined in TS 28.554. Performance metrics can

 also be those specified by other SDOs or vendor specific metrics.

 Performance metrics are identfied with their names. A name can also

 identify a vendor specific group of performance metrics.

 For measurements defined in TS 28.552 the name is constructed as

 follows:

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

 subcounters

 - 'family.measurementName' for measurement types without subcounters

 - 'family' for measurement families

 For KPIs defined in TS 28.554 the name is defined in the KPI

 definitions template as the component designated with e).";

 }

 leaf granularityPeriod {

 type uint32 {

 range 1..max ;

 }

 units seconds;

 mandatory true;

 description "Granularity period used to produce measurements. The value

 must be one of the supported granularity periods for the metric.";

 }

 leaf-list objectInstances {

 type types3gpp:DistinguishedName;

 }

 leaf-list rootObjectInstances {

 type types3gpp:DistinguishedName;

 description "Each object instance designates the root of a subtree that

 contains the root object and all descendant objects.";

 }

 uses types3gpp:ReportingCtrl {

 refine "reportingCtrl/file-based-reporting/fileReportingPeriod" {

 must '(number(.)\*"60") mod number(../granularityPeriod) = "0"' {

 error-message

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

 }

 }

 }

 leaf \_linkToFiles {

 type string;

 config false;

 mandatory true;

 yext3gpp:notNotifyable ;

 description "Link to a 'Files' object.";

 yext3gpp:inVariant;

 }

 }

 grouping ThresholdMonitorGrp {

 description "A threshold monitor that is created by the consumer for

 the monitored entities whose measurements are required by consumer

 to monitor.";

 leaf administrativeState {

 default UNLOCKED;

 type types3gpp:BasicAdministrativeState ;

 description "Enables or disables the ThresholdMonitor.";

 }

 leaf operationalState {

 config false;

 mandatory true;

 type types3gpp:OperationalState ;

 description "Indicates whether the ThresholdMonitor is working.";

 }

 list thresholdInfoList {

 key idx;

 min-elements 1;

 leaf idx { type uint32 ; }

 uses ThresholdInfoGrp;

 description "List of threshold info.";

 }

 leaf monitorGranularityPeriod {

 type uint32 {

 range "1..max";

 }

 units second;

 mandatory true;

 description "Granularity period used to monitor measurements for

 threshold crossings.";

 }

 leaf-list objectInstances {

 type types3gpp:DistinguishedName;

 yext3gpp:notNotifyable;

 }

 leaf-list rootObjectInstances {

 type types3gpp:DistinguishedName;

 description "Each object instance designates the root of a subtree that

 contains the root object and all descendant objects.";

 yext3gpp:notNotifyable;

 }

 }

 grouping MeasurementSubtree {

 description "Contains classes that define measurements.

 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!

 If a class uses this grouping in its list it shall also use the

 grouping SupportedPerfMetricGroupGrp to add SupportedPerfMetricGroup as

 an attribute to its grouping";

 list PerfMetricJob {

 description "This IOC represents a performance metric production job. It

 can be name-contained by SubNetwork, ManagedElement, or ManagedFunction.

 To activate the production of the specified performance metrics, a MnS

 consumer needs to create a PerfMetricJob instance on the MnS producer.

 For ultimate deactivation of metric production, the MnS consumer should

 delete the job to free up resources on the MnS producer.

 For temporary suspension of metric production, the MnS consumer can

 manipulate the value of the administrative state attribute. The MnS

 producer may disable metric production as well, for example in overload

 situations. This situation is indicated by the MnS producer with setting

 the operational state attribute to disabled. When production is resumed

 the operational state is set back to enabled.

 The jobId attribute can be used to associate metrics from multiple

 PerfMetricJob instances. The jobId can be included when reporting

 performance metrics to allow a MnS consumer to associate received

 metrics for the same purpose. For example, it is possible to configure

 the same jobId value for multiple PerfMetricJob instances required to

 produce the measurements for a specific KPI.

 The attribute performanceMetrics defines the performance metrics to be

 produced and the attribute granularityPeriod defines the granularity

 period to be applied.

 All object instances below and including the instance name-containing

 the PerfMetricJob (base object instance) are scoped for performance

 metric production. Performance metrics are produced only on those object

 instances whose object class matches the object class associated to the

 performance metrics to be produced.

 The attributes objectInstances and rootObjectInstances allow to restrict

 the scope. When the attribute objectInstances is present, only the object

 instances identified by this attribute are scoped. When the attribute

 rootObjectInstances is present, then the subtrees whose root objects are

 identified by this attribute are scoped. Both attributes may be present

 at the same time meaning the total scope is equal to the sum of both

 scopes. Object instances may be scoped by both the objectInstances and

 rootObjectInstances attributes. This shall not be considered as an error

 by the MnS producer.

 When the performance metric requires performance metric production on

 multiple managed objects, which is for example the case for KPIs, the

 MnS consumer needs to ensure all required objects are scoped. Otherwise

 a PerfMetricJob creation request shall fail.

 The attribute reportingCtrl specifies the method and associated control

 parameters for reporting the produced measurements to MnS consumers.

 Three methods are available: file-based reporting with selection of the

 file location by the MnS producer, file-based reporting with selection

 of the file location by the MnS consumer and stream-based reporting.

 For file-based reporting, all performance metrics that are produced

 related to a 'PerfMetricJob' instance for a reporting period shall be

 stored in a single reporting file.

 When the administrative state is set to 'UNLOCKED' after the creation

 of a 'PerfMetricJob' the first granularity period shall start. When

 the administrative state is set to 'LOCKED' or the operational state

 to 'DISABLED', the ongoing reporting period shall be aborted, for

 streaming the ongoing granularity period. When the administrative

 state is set back to 'UNLOCKED' or the operational state to 'ENABLED'

 a new reporting period period shall start, in case of streaming a new

 granularity period.

 Changes of all other configurable attributes shall take effect only at

 the beginning of the next reporting period, for streaming at the

 beginning of the next granularity period.

 When the 'PerfMetricJob' is deleted, the ongoing reporting period shall

 be aborted, for streaming the ongoing granularity period.

 A PerfMetricJob creation request shall fail, when the requested

 performance metrics, the requested granularity period, the requested

 repoting method, or the requested combination thereof is not supported

 by the MnS producer.

 Creation and deletion of PerfMetricJob instances by MnS consumers is

 optional; when not supported, PerfMetricJob instances may be created and

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

 key id;

 uses top3gpp:Top\_Grp ;

 container attributes {

 uses PerfMetricJobGrp ;

 }

 uses files3gpp:FilesSubtree {

 if-feature FilesUnderPerfMetricJob;

 }

 }

 list ThresholdMonitor {

 key id;

 description "Represents a threshold monitor for performance metrics.

 It can be contained by SubNetwork, ManagedElement, or ManagedFunction.

 A threshold monitor checks for threshold crossings of performance metric

 values and generates a notification when that happens.

 The ThresholdMonitor is used only when NRM based threshold

 monitoring is supported.

 To activate threshold monitoring, a MnS consumer needs to create a

 ThresholdMonitor instance on the MnS producer. For ultimate deactivation

 of threshold monitoring, the MnS consumer should delete the monitor to

 free up resources on the MnS producer.

 For temporary suspension of threshold monitoring, the MnS consumer can

 manipulate the value of the administrative state attribute. The MnS

 producer may disable threshold monitoring as well, for example in

 overload situations. This situation is indicated by the MnS producer with

 setting the operational state attribute to disabled. When monitoring is

 resumed the operational state is set again to enabled.

 All object instances below and including the instance containing the

 ThresholdMonitor (base object instance) are scoped for performance

 metric production. Performance metrics are monitored only on those

 object instances whose object class matches the object class associated

 to the performance metrics to be monitored.

 The optional attributes objectInstances and rootObjectInstances allow to

 restrict the scope. When the attribute objectInstances is present, only

 the object instances identified by this attribute are scoped. When the

 attribute rootObjectInstances is present, then the subtrees whose root

 objects are identified by this attribute are scoped. Both attributes may

 be present at the same time meaning the total scope is equal to the sum

 of both scopes. Object instances may be scoped by both the objectInstances

 and rootObjectInstances attributes. This shall not be considered as an

 error by the MnS producer.

 Multiple thresholds can be defined for multiple performance metric sets

 in a single monitor using thresholdInfoList. The attribute

 monitorGranularityPeriod defines the granularity period to be applied.

 The value is a multiple of a supported granularity period for the

 measurements being monitored.

 A threshold is defined using the attributes thresholdValue ,

 thresholdDirection and hysteresis.

 When hysteresis is absent or carries no information, a threshold is

 triggered when the thresholdValue is reached or crossed. When hysteresis

 is present, two threshold values are specified for the threshold as

 follows: A high treshold value equal to the threshold value plus the

 hysteresis value, and a low threshold value equal to the threshold value

 minus the hysteresis value. When the monitored performance metric

 increases, the threshold is triggered when the high threshold value is

 reached or crossed. When the monitored performance metric decreases, the

 threshold is triggered when the low threshold value is reached or crossed.

 The hsyteresis ensures that the performance metric value can oscillate

 around a comparison value without triggering each time the threshold when

 the threshold value is crossed. Using the thresholdDirection attribute a

 threshold can be configured in such a manner that it is triggered only

 when the monitored performance metric is going up or down upon reaching

 or crossing the threshold.

 A ThresholdMonitor creation request shall be rejected, if the performance

 metrics requested to be monitored, the requested granularity period, or

 the requested combination thereof is not supported by the MnS producer.

 A creation request may fail, when the performance metrics requested to be

 monitored are not produced by a PerfMetricJob.

 Creation and deletion of ThresholdMonitor instances by MnS consumers is

 optional; when not supported, ThresholdMonitor instances may be created

 and deleted by the system or be pre-installed.

 A threshold crossing event detected by a 'ThresholdMonitor' shall

 trigger a 'notifyThresholdCrossing' notification. To subscribe to

 'notifyThresholdCrossing' notifications the MnS consumer shall specify

 one or more 'ThresholdMonitor' instances in the subscription. All

 threshold crossings detected by the specified 'ThresholdMonitor'

 instances are sent as 'notifyThresholdCrossing' to subscribed MnS

 consumers (unless filtered out by the 'notificationFilter' attribute

 of 'NtfSubscriptionControl').";

 uses top3gpp:Top\_Grp ;

 container attributes {

 uses ThresholdMonitorGrp ;

 }

 }

 }

}

<CODE ENDS>

***Next change***

D.2.9 module \_3gpp-common-fm.yang

<CODE BEGINS>

module \_3gpp-common-fm {

 yang-version 1.1;

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

 prefix "fm3gpp";

 import ietf-yang-types { prefix yang; }

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

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

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

 organization "3GPP SA5";

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

 description "Defines a Fault Management model

 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 2024-02-23 { reference CR-0347; }

 revision 2023-09-17 { reference CR-0270; }

 revision 2023-05-09 { reference CR-0249; }

 revision 2022-10-31 { reference CR-0195; }

 revision 2021-08-08 { reference "CR-0132"; }

 revision 2021-06-02 { reference "CR-0130"; }

 revision 2020-06-03 { reference "CR-0091"; }

 revision 2020-02-24 { reference "S5-201365"; }

 feature AcknowledgeByConsumer {

 description "Indicates whether alarm acknowledgement by the consumer is

 supported.";

 }

 typedef eventType {

 type enumeration {

 enum COMMUNICATIONS\_ALARM {

 value 2;

 }

 enum QUALITY\_OF\_SERVICE\_ALARM {

 value 3;

 }

 enum PROCESSING\_ERROR\_ALARM {

 value 4;

 }

 enum EQUIPMENT\_ALARM {

 value 5;

 }

 enum ENVIRONMENTAL\_ALARM {

 value 6;

 }

 enum INTEGRITY\_VIOLATION {

 value 7;

 }

 enum OPERATIONAL\_VIOLATION {

 value 8;

 }

 enum PHYSICAL\_VIOLATION {

 value 9;

 }

 enum SECURITY\_SERVICE\_OR\_MECHANISM\_VIOLATION {

 value 10;

 }

 enum TIME\_DOMAIN\_VIOLATION {

 value 11;

 }

 }

 description "General category for the alarm.";

 }

 typedef severity-level {

 type enumeration {

 enum CRITICAL { value 3; }

 enum MAJOR { value 4; }

 enum MINOR { value 5; }

 enum WARNING { value 6; }

 enum INDETERMINATE { value 7; }

 enum CLEARED { value 8; }

 }

 description "The possible alarm severities";

 }

 grouping AlarmRecordGrp {

 description "Contains alarm information of an alarmed object instance.

 A new record is created in the alarm list when an alarmed object

 instance generates an alarm and no alarm record exists with the same

 values for objectInstance, alarmType, probableCause and specificProblem.

 When a new record is created the MnS producer creates an alarmId, that

 unambiguously identifies an alarm record in the AlarmList.

 Alarm records are maintained only for active alarms. Inactive alarms are

 automatically deleted by the MnS producer from the AlarmList.

 Active alarms are alarms whose

 a) perceivedSeverity is not CLEARED, or whose

 b) perceivedSeverity is CLEARED and its ackState is not ACKNOWLEDED.";

 leaf alarmId {

 type string;

 mandatory true;

 description "Identifies the alarmRecord";

 yext3gpp:notNotifyable;

 }

 leaf objectInstance {

 type types3gpp:DistinguishedName;

 config false ;

 mandatory true;

 yext3gpp:notNotifyable;

 yext3gpp:inVariant;

 }

 leaf notificationId {

 type int32;

 config false ;

 mandatory true;

 description "The Id of the last notification updating the AlarmRecord.";

 yext3gpp:notNotifyable;

 }

 leaf alarmRaisedTime {

 type yang:date-and-time ;

 mandatory true;

 config false ;

 yext3gpp:notNotifyable;

 }

 leaf alarmChangedTime {

 type yang:date-and-time ;

 config false ;

 description "not applicable if related alarm has not changed";

 yext3gpp:notNotifyable;

 }

 leaf alarmClearedTime {

 type yang:date-and-time ;

 config false ;

 description "not applicable if related alarm was not cleared";

 yext3gpp:notNotifyable;

 }

 leaf alarmType {

 type eventType;

 config false ;

 mandatory true;

 description "General category for the alarm.";

 yext3gpp:notNotifyable;

 yext3gpp:inVariant;

 }

 leaf probableCause {

 type union {

 type int32;

 type string;

 }

 config false ;

 mandatory true;

 yext3gpp:notNotifyable;

 yext3gpp:inVariant;

 }

 leaf specificProblem {

 type union {

 type int32;

 type string;

 }

 config false ;

 reference "ITU-T Recommendation X.733 clause 8.1.2.2.";

 yext3gpp:notNotifyable;

 yext3gpp:inVariant;

 }

 leaf perceivedSeverity {

 type severity-level;

 mandatory true;

 description "This is Writable only if producer supports consumer

 to set perceivedSeverity to CLEARED";

 yext3gpp:notNotifyable;

 }

 leaf backedUpStatus {

 type boolean;

 config false ;

 description "Indicates if an object (the MonitoredEntity) has a back

 up. See definition in ITU-T Recommendation X.733 clause 8.1.2.4.";

 yext3gpp:notNotifyable;

 }

 leaf backUpObject {

 type types3gpp:DistinguishedName;

 config false ;

 description "Backup object of the alarmed object as defined in

 ITU-T Rec. X. 733";

 yext3gpp:notNotifyable;

 }

 leaf trendIndication {

 type enumeration {

 enum MORE\_SEVERE;

 enum NO\_CHANGE;

 enum LESS\_SEVERE;

 }

 config false ;

 description "Indicates if some observed condition is getting better,

 worse, or not changing. ";

 reference "ITU-T Recommendation X.733 clause 8.1.2.6.";

 yext3gpp:notNotifyable;

 }

 grouping ThresholdInfoGrp {

 leaf measurementType {

 type string;

 mandatory true;

 }

 leaf direction {

 type enumeration {

 enum INCREASING;

 enum DECREASING;

 }

 mandatory true;

 description "

 If it is 'Increasing', the threshold crossing notification is

 triggered when the measurement value equals or exceeds a

 thresholdValue.

 If it is 'Decreasing', the threshold crossing notification is

 triggered when the measurement value equals or below a

 thresholdValue.";

 }

 leaf thresholdLevel {

 type string;

 }

 leaf thresholdValue {

 type string;

 }

 leaf hysteresis {

 type string;

 description "The hysteresis has a threshold high and a threshold

 low value that are different from the threshold value.

 A hysteresis, therefore, defines the threshold-high and

 threshold-low levels within which the measurementType value is

 allowed to oscillate without triggering the threshold crossing

 notification.";

 }

 }

 list thresholdInfo {

 config false ;

 yext3gpp:notNotifyable;

 description "Indicates the crossed threshold";

 uses ThresholdInfoGrp;

 }

 list stateChangeDefinition {

 key attributeName;

 config false ;

 description "Indicates MO attribute value changes associated with the

 alarm for state attributes of the monitored entity (state transitions).

 The change is reported with the name of the state attribute, the new

 value and an optional old value.

 See definition in ITU-T Recommendation X.733 [4] clause 8.1.2.10.";

 yext3gpp:notNotifyable;

 leaf attributeName {

 type string;

 }

 anydata newValue {

 mandatory true;

 description "The new value of the attribute. The content of this data

 node shall be in accordance with the data model for the attribute.";

 }

 anydata oldValue{

 description "The old value of the attribute. The content of this data

 node shall be in accordance with the data model for the attribute.";

 }

 }

 list monitoredAttributes {

 key attributeName;

 config false ;

 yext3gpp:notNotifyable;

 description "Attributes of the monitored entity and their

 values at the time the alarm occurred that are of interest for the

 alarm report.";

 reference "ITU-T Recommendation X.733 clause 8.1.2.11.";

 leaf attributeName {

 type string;

 }

 anydata value {

 mandatory true;

 description "The value of the attribute. The content of this data

 node shall be in accordance with the data model for the attribute.";

 }

 }

 leaf proposedRepairActions {

 type string;

 config false ;

 description "Indicates proposed repair actions. See definition in

 ITU-T Recommendation X.733 clause 8.1.2.12.";

 yext3gpp:notNotifyable;

 }

 leaf additionalText {

 type string;

 config false ;

 yext3gpp:notNotifyable;

 }

 list additionalInformation {

 key name;

 config false ;

 yext3gpp:notNotifyable;

 description "Vendor specific alarm information in the alarm.";

 uses types3gpp:nameValuePair;

 }

 leaf rootCauseIndicator {

 type boolean;

 default false;

 config false ;

 description "It indicates that this AlarmInformation is the root cause

 of the events captured by the notifications whose identifiers are in

 the related CorrelatedNotification instances.";

 yext3gpp:notNotifyable;

 }

 leaf ackTime {

 if-feature AcknowledgeByConsumer;

 type yang:date-and-time ;

 config false ;

 description "It identifies the time when the alarm has been

 acknowledged or unacknowledged the last time, i.e. it registers the

 time when ackState changes.";

 yext3gpp:notNotifyable;

 }

 leaf ackUserId {

 if-feature AcknowledgeByConsumer;

 type string;

 description "It identifies the last user who has changed the

 Acknowledgement State.";

 yext3gpp:notNotifyable;

 }

 leaf ackSystemId {

 if-feature AcknowledgeByConsumer;

 type string;

 description "It identifies the system (Management System) that last

 changed the ackState of an alarm, i.e. acknowledged or unacknowledged

 the alarm.";

 yext3gpp:notNotifyable;

 }

 leaf ackState {

 if-feature AcknowledgeByConsumer;

 type enumeration {

 enum ACKNOWLEDGED {

 description "The alarm has been acknowledged.";

 }

 enum UNACKNOWLEDGED {

 description "The alarm has unacknowledged or the alarm has never

 been acknowledged.";

 }

 }

 yext3gpp:notNotifyable;

 }

 leaf clearUserId {

 type string;

 description "Carries the identity of the user who invokes the

 clearAlarms operation.";

 yext3gpp:notNotifyable;

 }

 leaf clearSystemId {

 type string;

 yext3gpp:notNotifyable;

 }

 leaf serviceUser {

 type string;

 config false ;

 description "It identifies the service-user whose request for service

 provided by the serviceProvider led to the generation of the

 security alarm.";

 yext3gpp:notNotifyable;

 }

 leaf serviceProvider {

 type string;

 config false ;

 description "It identifies the service-provider whose service is

 requested by the serviceUser and the service request provokes the

 generation of the security alarm.";

 yext3gpp:notNotifyable;

 }

 leaf securityAlarmDetector {

 type string;

 config false ;

 yext3gpp:notNotifyable;

 }

 list correlatedNotifications {

 key sourceObjectInstance;

 description "List of correlated notifications";

 leaf sourceObjectInstance {

 type types3gpp:DistinguishedName;

 }

 leaf-list notificationId {

 type int32;

 min-elements 1;

 }

 }

 }

 grouping AlarmListGrp {

 description "Represents the AlarmList IOC.";

 leaf administrativeState {

 type types3gpp:BasicAdministrativeState ;

 default LOCKED;

 description "When set to UNLOCKED, the alarm list is updated.

 When the set to LOCKED, the existing alarm records are not

 updated, and new alarm records are not added to the alarm list.";

 }

 leaf operationalState {

 type types3gpp:OperationalState ;

 default DISABLED;

 config false;

 description "The producer sets this attribute to ENABLED, indicating

 that it has the resource and ability to record alarm in AlarmList

 else, it sets the attribute to DISABLED.";

 }

 leaf numOfAlarmRecords {

 type uint32 ;

 config false;

 mandatory true;

 description "The number of alarm records in the AlarmList";

 yext3gpp:notNotifyable;

 }

 leaf lastModification {

 type yang:date-and-time ;

 config false;

 description "The last time when an alarm record was modified";

 yext3gpp:notNotifyable;

 }

 list alarmRecords {

 key alarmId;

 description "List of alarmRecords";

 yext3gpp:notNotifyable;

 uses AlarmRecordGrp;

 }

 }

 grouping FmSubtree {

 description "Contains FM related classes.

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

 - SubNetwork

 - ManagedElement

 If some YAM wants to augment these classes/list/groupings they must

 augment all user classes!";

 list AlarmList {

 key id;

 max-elements 1;

 description "The AlarmList represents the capability to store and manage

 alarm records. The management scope of an AlarmList is defined by all

 descendant objects of the base managed object, which is the object

 name-containing the AlarmList, and the base object itself.

 AlarmList instances are created by the system or are pre-installed.

 They cannot be created nor deleted by MnS consumers.

 When the alarm list is locked or disabled, the existing alarm records

 are not updated, and new alarm records are not added to the alarm list";

 uses top3gpp:Top\_Grp ;

 container attributes {

 uses AlarmListGrp ;

 }

 }

 }

}

<CODE ENDS>

***End of changes***